PREFACE

The IBM Personal Computer Technical Reference Manual is designed to provide hardware design and interface information. This publication also provides Basic Input Output System (BIOS) information as well as programming support matter.

This manual is intended for programmers, engineers involved in hardware and software design, designers, and interested persons who have a need to know how the IBM Personal Computer is designed and works.

This manual has three sections:

Section-1

"HARDWARE OVERVIEW," features an overview of the system as a whole calling out specific items such as the System Unit, Keyboard, IBM Monochrome Display and the 80 CPS Graphics Printer.

Section-2

"HARDWARE," contains a description for each functional part of the system. This section also contains specifications for power, timing, and interface. Programming considerations are supported by coding tables, command codes and registers.

Section-3

"ROM and SYSTEM USAGE," describes BIOS as well as how to use BIOS, interrupt vector listings, memory map, vectors with special meanings, a cassette section, a keyboard encoding section, and a set of Low Memory Maps.

"APPENDICES," to address the ROM BIOS listing, an instruction set, logic diagrams, and expanded charts used to support specific hardware descriptions.

IBM

Personal Computer Hardware Reference Library

Technical Reference

CONTENTS

FIGURE LISTING

TABLE LISTING

SECTION 1.
HARDWARE OVERVIEW

The IBM Personal Computer has two major elements; a System Unit and a keyboard. In addition, a variety of options are offered including one or two 5-1/4" Diskette Drives with adapter which can be housed inside the System Unit, an IBM Monochrome Display, an IBM 80 CPS Graphics Printer, two display adapters, storage increments to 544KB, an Asynchronous Communications Adapter, Printer Adapter and a Game Control Adapter.

The System Unit is the heart of your IBM Personal Computer system. The System Unit houses the microprocessor, Read-Only Memory (ROM), Read/Write Memory, Power Supply, and System Expansion Slots for the attachment of up to five options. One or two 5-1/4" Diskette Drives can also be mounted in the System Unit providing 160KB or 320 KB of storage each, depending on the type of drive you have.

The System Board is a large board which fits horizontally in the base of the System Unit and includes the microprocessor, 40KB ROM and 16KB memory. The memory can be expanded in 16KB increments to 64KB. The System Board also includes an enhanced version of the Microsoft BASIC-80 Interpreter without diskette functions. The BASIC Interpreter is included in the ROM. The System Board also permits the attachment of an audio cassette recorder for loading or saving programs and data.

The System Unit power supply is a 63.5 watt, 4 level DC and either 120 Vac or 220/240 Vac unit. It is a switching regulator design, allowing for light weight and high efficiency. Its DC power capacity is rated for an expanded system.

The 5-1/4" Diskette Drive permits the IBM Personal Computer to read, write and store data on 5-1/4" diskettes. Each diskette stores approximately 160KB or 320KB of formatted data. Two of these drives may be installed internally in the System Unit.

The keyboard is attached to the System Unit with a light-weight, coiled cable. The keyboard features 83 keys, and offers commonly used data and word processing functions in a design combining the familiar typewriter and calculator pad layouts.

A base system requires one of two different display adapters, either a Color/Graphics medium resolution Monitor adapter or a high resolution monochrome alphanumeric adapter with a parallel printer adapter.

The Color/Graphics Adapter operates at U.S. NTSC (National Television System Committee) standard frequencies (15,750 Hz horizontal and 60 Hz vertical scan), allowing direct attachment to a variety of home TVs (with a user supplied RF modulator) and monitors.

The Color/Graphic Monitor adapter supports a variety of modes selected by program control. The adapter supports color or black and white alphanumeric modes with line width of 40 or 80 characters and 25 lines. In the alphanumeric mode there are 256 characters.

This adapter provides both a standard composite video and direct drive outputs. In addition, a light pen feature input port is provided.

The IBM Monochrome Display is a high resolution green phosphor display offering the personal computer user quality usually found on larger computer systems. The display features an 11-1/2" screen with an anti-glare surface and a variety of highlighting choices. The screen displays 25 lines of 80 characters. It supports 256 different letters, numbers and special characters that are formed in a nine by 14 dot matrix.

The IBM Monochrome Display requires the Monochrome Display and Printer Adapter Option. This option installs in one of the System Unit's five System Expansion Slots. The display is powered from the System Unit.

The IBM 80 CPS Graphics Printer is a versatile, low cost, quality printer. It prints in both directions at a nominal horizontal speed of 80 characters per second on continuous-feed, single or multi-part paper. The printer features four character sizes (40, 66, 80, or 132 characters per line), subscript, superscript, and underlining. It has uppercase and lowercase ASCII and International character sets, a defined graphic character set and programmable graphics. In addition it has a Power-on Self-test, simple paper loading and a cartridge ribbon.

The IBM 80 CPS Graphics Printer requires either the Monochrome Display and Printer Adapter or the Printer Adapter installed in one of the System Unit's five System Expansion Slots. The printer is available in either a 120 volt, 60 Hz version or a 220 or 240 volt, 50/60 Hz version and plugs into a standard wall outlet. The printer requires the Printer Cable Option for attachment to the System Unit.

The 16KB Memory Module Kits allow you to increase the memory size of your IBM Personal Computer. The base system comes standard with 16KB of memory. Up to three 16KB Memory Module Kits may be installed to increase the memory size to 64KB. Memory can be further increased to 544KB with additional memory options once these three Expansion Kits are installed.

The 16KB Memory Kits plug into the System Board and must be installed sequentially. They do not occupy any of the five System Expansion Slots.

The 32KB, 64KB, and 64/256KB Memory Expansion Options permit memory capacity to be increased beyond 64KB. Any combination of up to three Memory Expansion Options may be installed as long as System Expansion Slots are available. A maximum of 480KB of Memory Expansion Options can be addressed, which when added to 64KB on the System Board, gives a total memory capacity of 544KB.

Each 32KB, 64KB, or 64/256KB Memory Expansion Option requires a System Expansion Slot in the System Unit. The first 64KB on the System Board is required before any Memory Expansion Options can be installed.

The 64KB Memory Module Kits allow the memory size of a 64/256KB Memory Expansion Option to be increased. Each 64KB Kit consists of 9 modules which plug into the 64/256KB Option and must be installed sequentially. The base 64/256KB Option comes standard with 64KB of memory. One, two, or three 64KB Kits may be added, providing the 64/256KB Option with 128KB, 192KB, or 256KB of memory respectively.

The Asynchronous Communications Adapter provides a channel to data processing or input/output devices outside of your immediate system. These can be connected by telephone using a plug-in modem, or directly by cable when the device is nearby.

This option utilizes the RS232C asynchronous (start-stop) interface permitting attachment to a variety of devices including a large "host" computer or another IBM Personal Computer.

This option supports 50 to 9600 BPS transmission speeds. One 25 pin "D" shell, male type connector is provided to attach various peripheral devices. A "current loop" interface is located in the same connector, and a jumper block is provided to manually select either the voltage or the current loop interface.

The Asynchronous Communications Adapter requires a System Expansion Slot in the System Unit. Telephone line transmission requires an external modem which is subject to local telephone administration and national regulations. Note that references to local telephone administration and national regulations should not be taken to imply that permission to connect has been, or will be, obtained in any particular country.

The Game Control Adapter permits the attachment of user-supplied Joysticks or paddles. Two Joysticks and up to four paddles may be attached. IBM does not manufacture either the Joysticks or the paddles. This option provides connectors for Joysticks or paddles and requires a System Expansion Slot in the System Unit.

A block diagram of the system is on the following pages (1-6 and 1-7).

The Prototype Card is provided by IBM for those who wish to design a customized option for the IBM Personal Computer. The Prototype Card is inserted into an expansion slot in the System Unit.

System Block Diagram

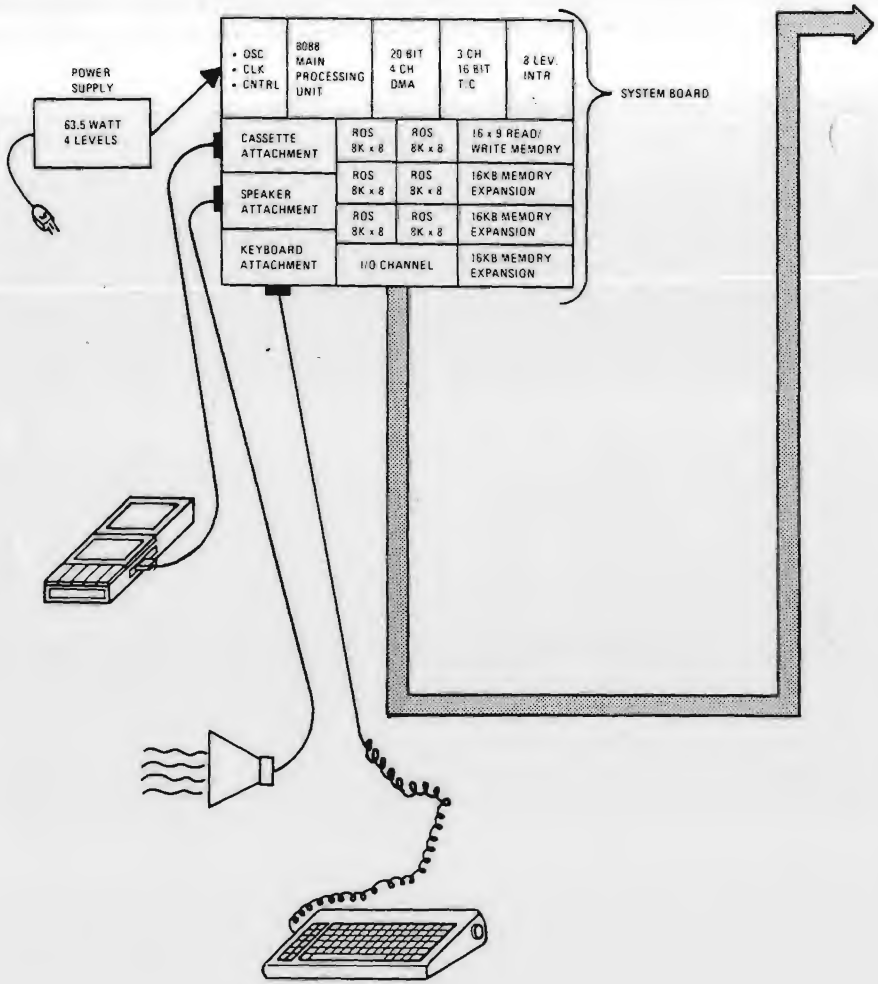


Figure 1. SYSTEM BLOCK DIAGRAM

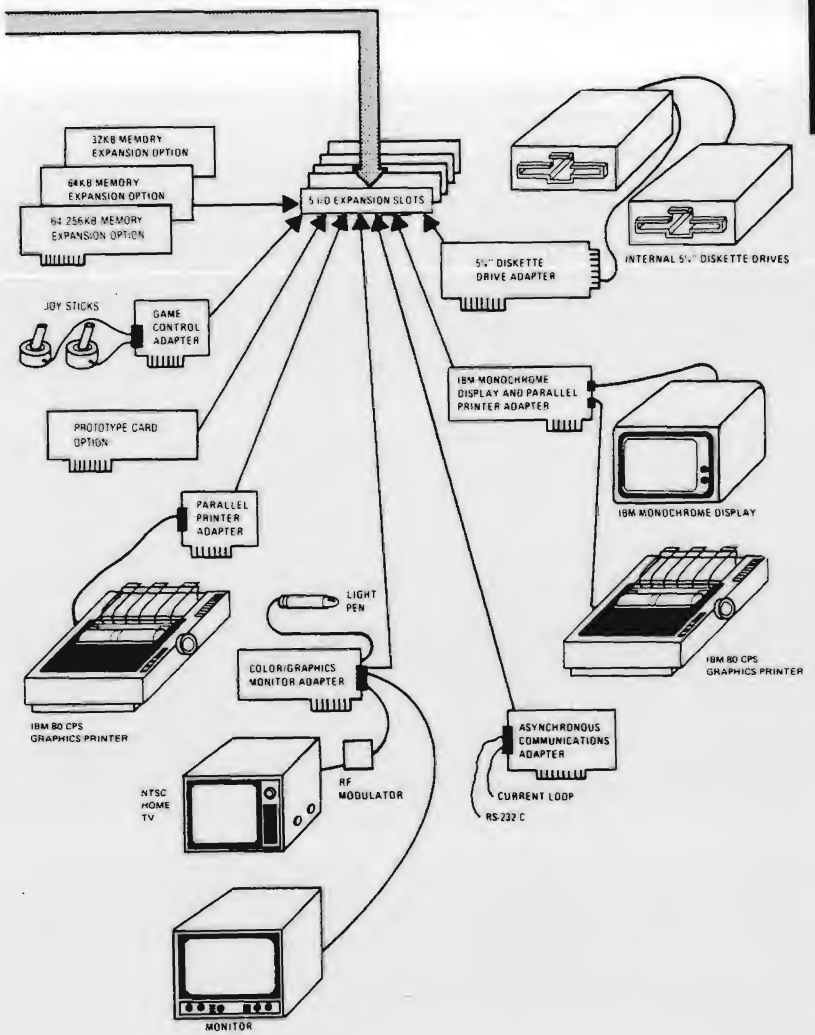


Figure 1. SYSTEM BLOCK DIAGRAM (continued)

System Board

The System Board fits horizontally in the base of the System Unit and has dimensions of approximately 8-1/2 inches by 11 inches. The System Board is a multilayer single land-per-channel design, with ground and power internal planes provided. DC power and a signal from the power supply enter the board through two six pin connectors. Other connectors on the board are for attaching the keyboard, audio cassette, and the speaker. Five 62-pin card edge sockets are also mounted on the System Board. The system I/O channel is bussed across these five I/O slots.

There are 16 (13 used) Dual In-line Package (DIP) switches mounted on the card which can be read under program control. These switches are used to indicate to the system software what options are installed. They are used to indicate amounts of installed storage, both on the System Board and in the System Expansion slots, type of display adapter installed, and desired operation modes upon power-up; ie, color or black and white and 80- or 40 character lines. Switches are also used to identify when the operating system is to be loaded from diskette, and how many diskette drives are attached.

The major elements of the System Board are divided into five major functional areas. They are, the processor subsystem and its support elements, the Read-Only Memory (ROM) subsystem, the Read/Write (R/W) Memory subsystem, integrated I/O adapters, and the I/O channel. All functions are described in detail in this section, except for the I/O channel, which has its own section. Figure 2.0 "System Board Data Flow" page 2-6, illustrates these functional areas.

The heart of the System Board is the Intel 8088 microprocessor. This processor is an 8-bit bus version of the 16-bit 8086 processor by Intel. It is software compatible with the 8086 and, thus, supports 16-bit operations including multiply and divide. The processor supports 20 bits of addressing (1 megabyte of storage). The processor is implemented in maximum mode so a co-processor can be added as a feature. The processor is operated at 4.77 Mhz. This frequency is derived from a 14.31818 Mhz crystal which is divided by three for the processor clock and by four to obtain the 3.58 Mhz color burst signal required for color televisions.

At the 4.77 Mhz clock rate, the 8088 bus cycles are four clocks of 210 ns or 840 ns. I/O cycles take five 210 ns clocks or 1.05 microsec (m sec).

SECTION 2. HARDWARE

Contents

The processor is supported by a set of high function support devices providing four channels of 20-bit Direct Memory Access (DMA), three 16-bit timer counter channels, and eight prioritized interrupt levels.

Three of the four DMA channels are available on the I/O bus and are provided to support high speed data transfers between I/O devices and memory without processor intervention. The fourth DMA channel is programmed to refresh the system dynamic memory. This is done by programming a channel of the timer counter device to periodically request a dummy DMA transfer. This creates a memory read cycle which is available to refresh dynamic storage both on the System Board and in the System Expansion slots. All DMA data transfers, except the refresh channel, take five processor clocks of 210 ns or 1.05 ns if the processor ready line is not deactivated. Refresh DMA cycles take four clocks or 840 ns.

The three timer/counters are used by the system as follows: Channel 0 is used to time and request refresh cycles from the DMA channel, Channel 2 is used to support the tone generation for the audio speaker, and Channel 1 is used by the system as a general purpose timer providing a constant time base for implementing a time-of-day clock. Each channel has a minimum timing resolution of 1.05 μsec.

Of the eight prioritized levels of interrupt, six are bussed to the I/O slots for use by feature cards. Two levels are used on the System Board. Level 0, the highest priority, is attached to Channel 1 of the timer counter and provides a periodic interrupt. Level 1 is attached to the keyboard adapter circuits and receives an interrupt for each scan code sent by the keyboard. The Non-Maskable Interrupt (NMI) of the 8088 is used to report memory parity errors.

The System Board is designed to support both ROM and Read/Write Memory. The System Board contains space for 48K x 8 of ROM or EPROM. Six module sockets are provided, each capable of accepting an 8K x 8 device. Five of the sockets are populated with 40KB of ROM. This ROM contains the Cassette BASIC interpreter, cassette operating system, Power-on Self-test, I/O drivers, dot patterns for 128 characters in graphics mode, and a diskette bootstrap loader. The ROM is packaged in 24-pin modules and has an access time of 250ns and a cycle time of 375 ns.

The System Board also contains from 16K x 9 to 64K x 9 of Read/Write Memory. A minimum system would have 16KB of memory with module sockets for an additional 48KB. In a cassette version of the system, approximately 4KB is used by the system leaving approximately 12KB of user's space for BASIC programs. Additional memory beyond the System Board's maximum of 64KB, is obtained by adding memory cards in the System Expansion slots.

The memory is dynamic 16K x 1 chips with an access time of 250 ns and a cycle time of 410 ns. All R/W memory is parity checked.

The System Board contains circuits for attaching an audio cassette, the serial keyboard, and the speaker. The cassette adapter allows the attachment of any good quality audio cassette via either the microphone or auxiliary inputs. The board has a jumper for either input. This interface also provides a cassette motor control line for transport starting and stopping under program control. This interface reads and writes the audio cassette at a data rate of between 1,000 and 2,000 baud. The baud rate is variable and dependent on data content since a different bit-cell time is used for 0's and 1's. For diagnostic purposes, the tape interface can loop read to write to test the board's circuits. The system software blocks cassette data, generates and checks data with a Cyclic Redundancy Check (CRC).

The processor also contains the adapter circuits for attaching the serial interface from the keyboard. This generates an interrupt to the processor when a complete scan code is received. This interface can request execution of a diagnostic in the keyboard.

Both the keyboard and cassette interfaces are provided via 5-pin DIN connectors, which are right angle mounts on the System Board and extend through the rear panel of the System Unit.

The system is provided with a 2-1/4 inch audio speaker mounted inside the System Unit. The System Board contains the control circuits and driver for the speaker. The speaker connects through a 2-wire interface which attaches to a 4-pin header on the System Board.

The speaker drive circuit is capable of approximately a 1/2 watt of power. The control circuits allow the speaker to be driven several different ways. First, a direct program control register bit may be toggled to generate a pulse train; second, the output of Channel 2 of the timer counter may be programmed to generate a waveform to the speaker. Third, the clock input to the timer/counter can be modulated with a program controlled I/O Register Bit. All three forms of control may be performed simultaneously.

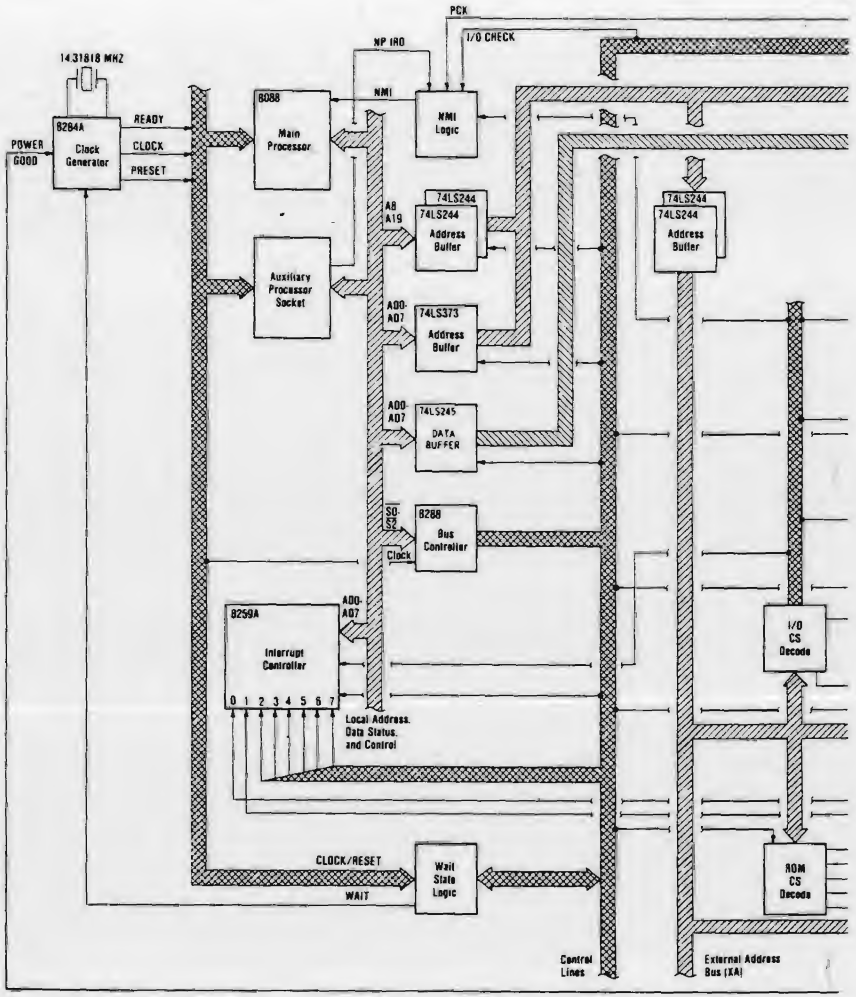


Figure 2. SYSTEM BOARD DATA FLOW (SHEET 1 OF 2)

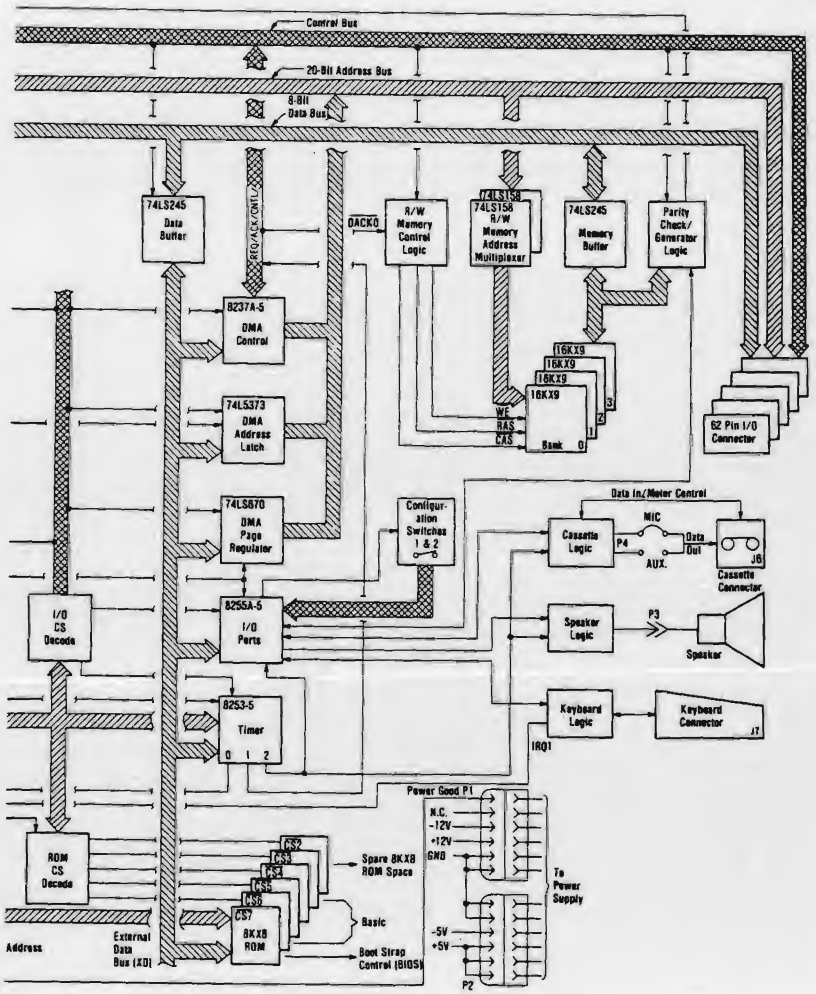


Figure 2. SYSTEM BOARD DATA FLOW (SHEET 2)

I/O Channel

The I/O channel is an extension of the 8088 microprocessor bus. It is, however, demultiplexed, repowered, and enhanced by the addition of interrupts and Direct Memory Access (DMA) functions.

The I/O channel contains an 8-bit bidirectional data bus, 20 address lines, 6 levels of interrupt, control lines for memory and I/O read or write, clock and timing lines, 3 channels of DMA control lines, memory refresh timing control lines a channel check line, and power and ground for the adapters. Four voltage levels are provided for I/O card +5 Vdc, -5 Vdc, +12 Vdc, and -12 Vdc. These functions are provided in a 62-pin connector with 100 mil card tab spacing.

A ready line is available on the I/O channel to allow operation with slow I/O or memory devices. If the channel's Ready line is not activated by an addressed device, all processor generated memory read and write cycles take four 210 ns clock or 840 ns/byte. All processor-generated I/O read and write cycles require five 210 ns clocks or 1.05 m sec/byte. All DMA transfers require five clocks for a cycle time of 1.05 m sec/byte. Refresh cycles are present once every 72 clocks or approximately 15 m sec and require five clocks or approximately 7% of the bus bandwidth.

I/O devices are addressed using I/O mapped address space. The channel is designed so that 512 I/O device addresses are available to the I/O channel cards.

A channel check line exists for reporting error conditions to the processor. Activating this line results in a NMI to the 8088 processor. Memory Expansion Options use this line to report parity errors.

The I/O channel is repowered so there is sufficient drive to power all five System Expansion Slots, assuming two loads per slot. The IBM Option I/O adapters typically use only one load. A graphic illustration of the System I/O Channel and its descriptions are on the following pages.

I/O Channel Diagram

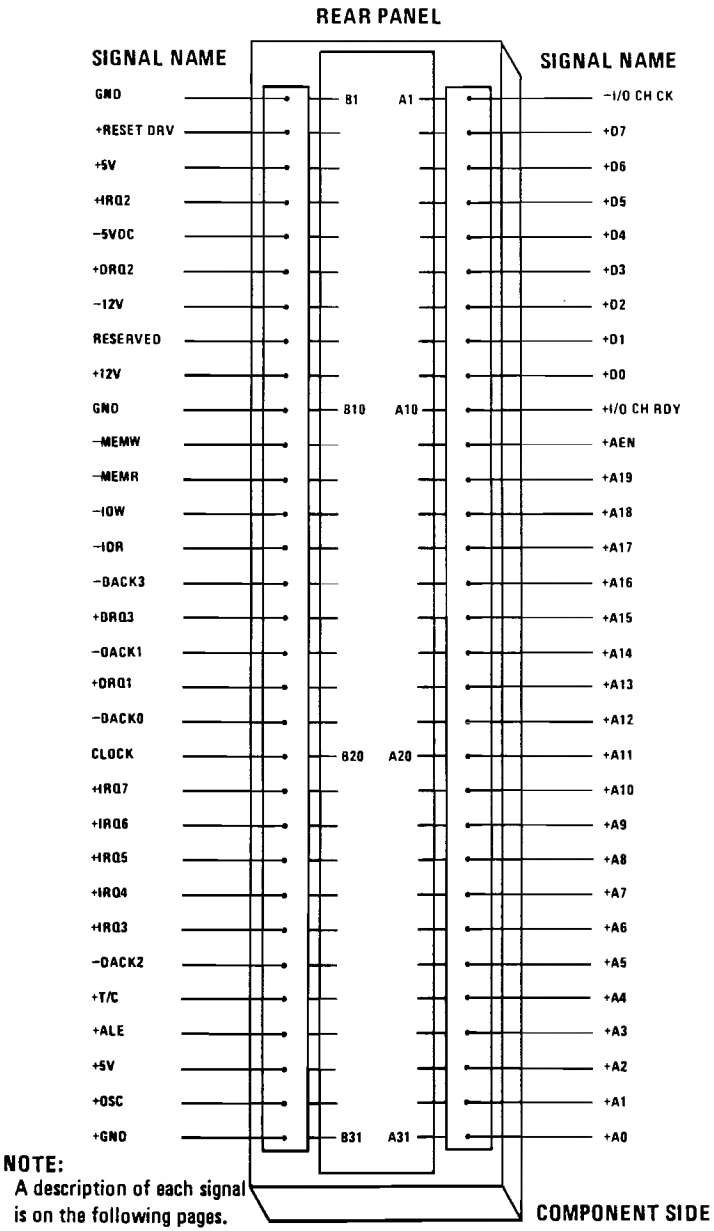


Figure 3. I/O CHANNEL DIAGRAM

System Board I/O Channel Description

The following is a description of the IBM Personal Computer System Board I/O Channel. All signal lines are TTL compatible.

Signal	I/O	Description
OSC	O	Oscillator: This signal is a high speed clock with a 70 nsec. period (14.31818 MHz). It has a 50% duty cycle.
CLK	O	Clock: This is the system clock. It is a divide - by - three of the oscillator and has a period of 210 nsec. (4.77 Mhz.) The clock has a 33% duty cycle.
RESET DRV	O	Reset Driver: This line is used to reset or initialize system logic upon power-up or during a low line voltage outage. This signal is synchronized to the falling edge of clock and is active HIGH.
A0-A19	O	Address Bits 0 to 19: These lines are used to address memory and I/O devices within the system. The 20 address lines allow access of up to 1 megabyte of memory. A0 is the Least Significant Bit (LSB) while A19 is the Most Significant Bit (MSB). These lines are generated by either the processor or the DMA Controller. They are active HIGH.
D0-D7	I/O	Data Bits 0 to 7: These lines provide data bus bits 0 to 7 for the processor, memory, and I/O Devices. D0 is the Least Significant Bit (LSB) and D7 is the Most Significant Bit (MSB). These lines are active HIGH.
ALE	O	Address Latch Enable: This is provided by the 8288 Bus Controller and is used on the System Board to latch valid addresses from the processor. It is available to the I/O Channel as an indicator of a valid processor address (when used in conjunction with AEN). Processor addresses are latched with the falling edge of ALE.
$\overline{\text{I/O CH CK}}$	I	-I/O Channel Check: This line provides the CPU with parity (error) information on memory or devices in the I/O Channel. When this signal is active LOW, a parity error is indicated.

Signal	I/O	Description
I/O CH RDY	I	I/O Channel Ready: This line (normally high or "READY") is pulled low ("NOT READY") by a memory or I/O device to lengthen I/O or memory cycles. It allows slower devices to attach to the I/O Channel with a minimum of difficulty. Any slow device using this line should drive it low immediately upon detecting a valid address and a Read or Write command. This line should never be held low for any period in excess of 10 clock cycles (2.1 usec.) Machine cycles (I/O or memory) are extended by an integral number of CLK cycles (210 ns).
IRQ2-IRQ7	I	Interrupt Request 2 to 7: These lines are used to signal the processor that an I/O device requires attention. They are prioritized with IRQ2 as the highest priority and IRQ7 as the lowest. An Interrupt Request is generated by raising an IRQ line (Low to High) and holding it high until it is acknowledged by the processor (Interrupt Service Routine).
$\overline{\text{IOR}}$	O	-I/O Read Command: This command line instructs an I/O device to drive its data onto the data bus. It may be driven by the processor or the DMA Controller. This signal is active LOW.
$\overline{\text{IOW}}$	O	-I/O Write Command: This command line instructs an I/O device to read the data on the data bus. It may be driven by the processor or the DMA Controller. This signal is active LOW.
$\overline{\text{MEMR}}$		-Memory Read Command: This command line instructs the memory to drive its data onto the data bus. It may be driven by the processor or the DMA Controller. This signal is active LOW.
$\overline{\text{MEMW}}$	O	-Memory Write Command: This command line instructs the memory to store the data present on the data bus. It may be driven by the processor or the DMA Controller. This signal is active LOW.

Signal	I/O	Description
DRQ1-DRQ3	I	DMA Request 1 to 3: These lines are asynchronous channel requests used by peripheral devices to gain DMA service. They are prioritized with DRQ1 having highest priority and DRQ3 the lowest. A request is generated by bringing a DRQ line to an active level (HIGH). A DRQ line must be held high until the corresponding DACK line goes active.
$\overline{\text{DACK0-DACK3}}$	O	-DMA Acknowledge 0 to 3: These lines are used to acknowledge DMA requests (DRQ1-DRQ3) and to refresh system dynamic memory (DACK0). They are active LOW.
AEN	O	Address Enable: This line is used to degate the processor and other devices from the I/O Channel to allow Direct Memory Access (DMA) transfers to take place. When this line is active (HIGH), the DMA Controller has control of the address bus, data bus, read command lines, (memory and I/O), and the write command lines, (memory and I/O).
T/C	O	Terminal Count: This line provides a pulse when the terminal count for any DMA channel is reached. This signal is active HIGH.

The following voltages are available on the System Board I/O Channel:

+5 Vdc ± 5%, Located on 2 connector pins.

-5 Vdc ± 10%, Located on 1 connector pin.

+12 Vdc ± 5%, Located on 1 connector pin.

-12 Vdc ± 10%, Located on 1 connector pin.

GND (Ground), Located on 3 connector pins.

System Board Component Diagram

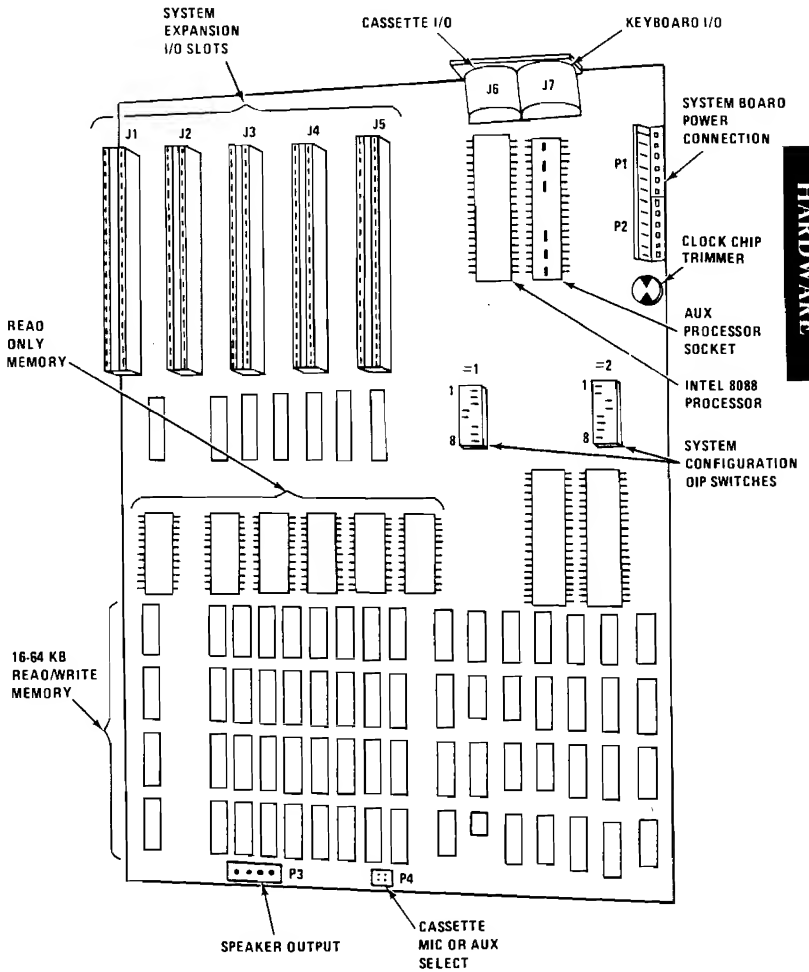


Figure 4. SYSTEM BOARD COMPONENT DIAGRAM

Keyboard

The Keyboard is a device separate from the System Unit. It is attached via a serial interface cable approximately 6 feet in length which plugs into the rear of the System Unit. The attaching cable is coiled, like that of a telephone headset, and is a shielded four-wire wire connection. The interface contains power (+5 Vdc), ground and two bidirectional signal lines. The cable is permanently attached at the keyboard end and plugs into the System Unit via a DIN connector.

The keyboard uses a capacitive technology with a microcomputer (Intel 8048) performing the keyboard scan function. The keyboard interface is defined so system software has the maximum flexibility in defining keyboard operations such as shift states of keys, make/break keys, and typematic operation. This is accomplished by having the keyboard return scan codes rather than American Standard Code for Information Interchange (ASCII) codes. In addition, all keys except control keys are typematic and generate both a make and a break scan code. For example, key 1 produces scan code 01 on make, and code 81 on break. Break codes are formed by adding X'80' to make codes. The keyboard I/O driver can define keyboard keys as shift or typematic keys as required by the application.

The microcomputer (Intel 8048) in the keyboard performs several functions including a Power-on Self-test and when requested by the System Unit. This diagnostic CRC checks the micro-computer ROM, tests memory and checks for stuck keys. Additional functions are: keyboard scanning, key debounce, buffering of up to 20 key scan codes, maintaining bidirectional serial communications with the System Unit, and executing the hand shake protocol required by each scan code transfer. A keyboard diagram and table of scan codes are on the following pages. Figure (5) is a block diagram of the keyboard interface on the System Board.

Several different keyboard arrangements are available. These are illustrated on the following pages together with a table of scan codes for the U.S. keyboard. For information on the keyboard routines required to implement non-U.S. keyboards, refer to the Guide to Operations and DOS manuals.

Note: Reference to local telephone administration and national regulations should not be taken to imply that permission to connect has been, or will be, obtained in any particular country.

Keyboard Interface Block Diagram

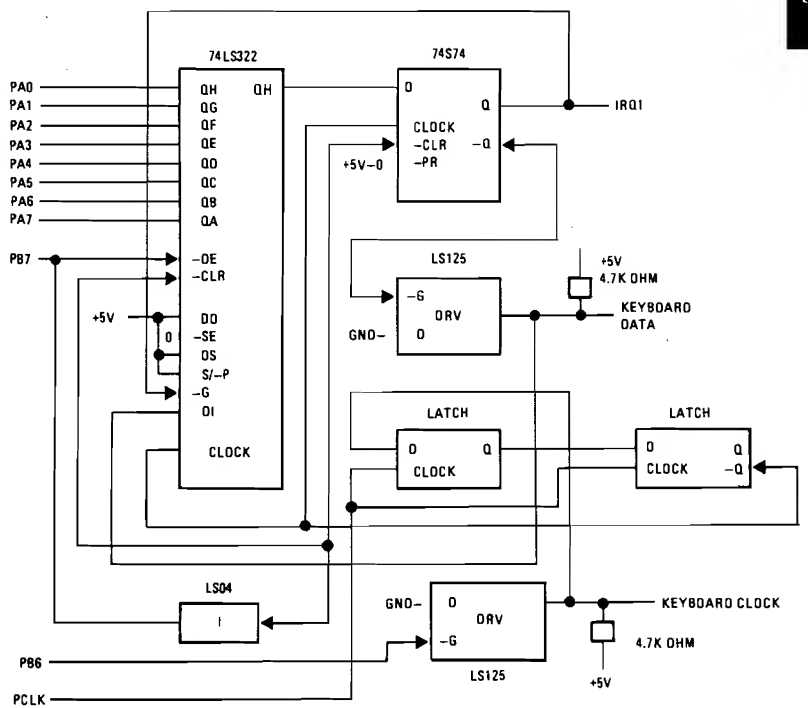
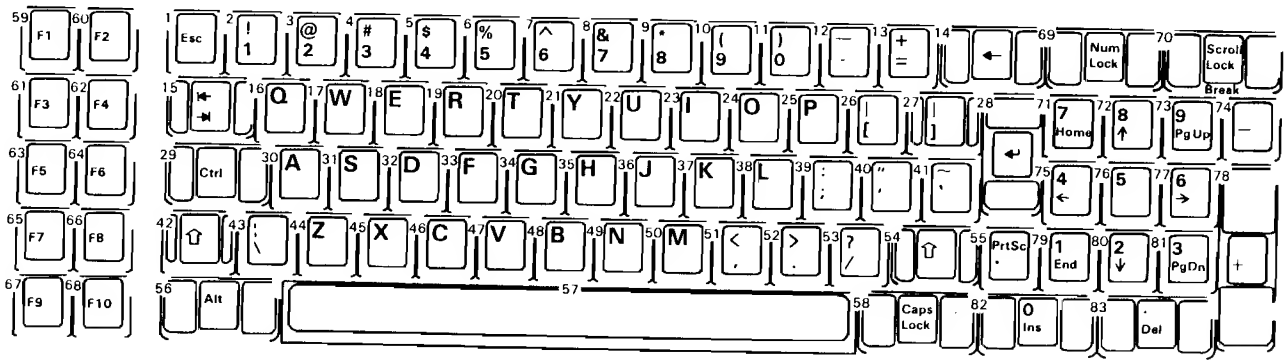
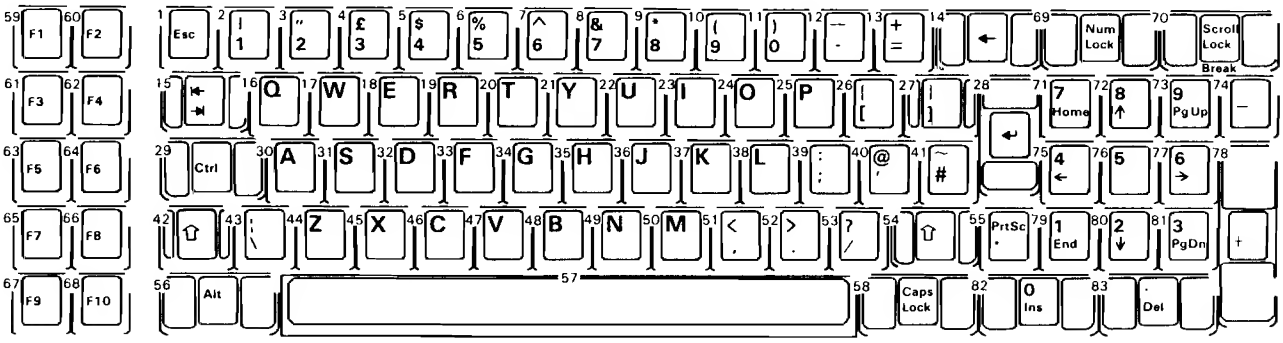


Figure 5. KEYBOARD INTERFACE BLOCK DIAGRAM



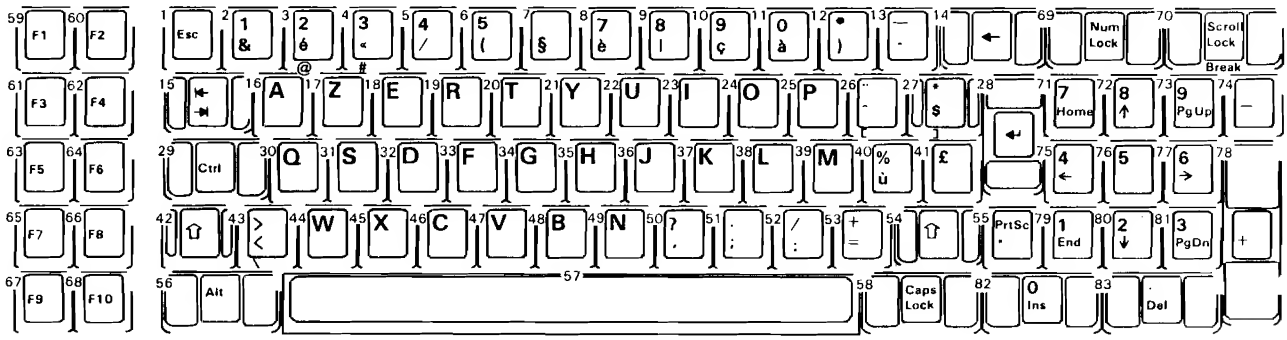
NOTE
1 NOMENCLATURE IS ON BOTH TOP AND FRONT FACE OF KEYBUTTON AS SHOWN. THE NUMBER TO THE UPPER LEFT DESIGNATES THE BUTTON POSITION.

Figure 6. KEYBOARD DIAGRAM



NOTE
1 NOMENCLATURE IS ON BOTH TOP AND FRONT FACE OF KEYBUTTON AS SHOWN. THE NUMBER TO THE UPPER LEFT DESIGNATES THE BUTTON POSITION.

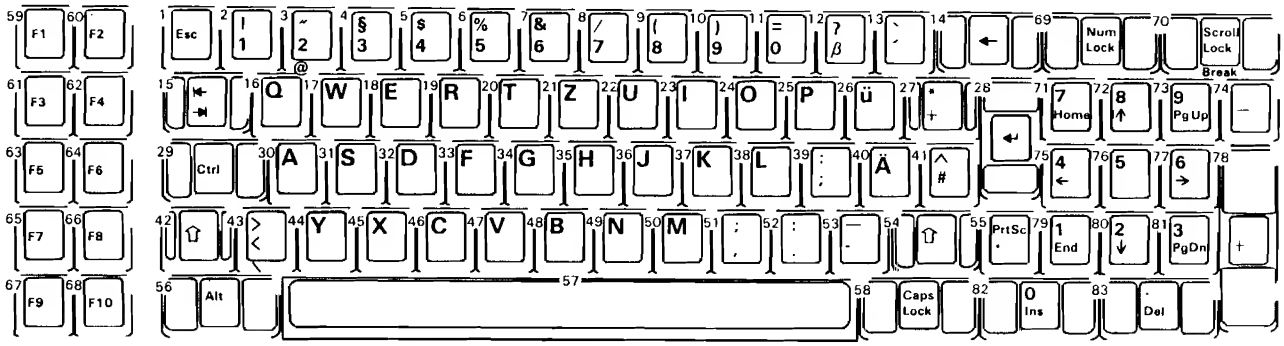
Figure 6. KEYBOARD DIAGRAM (Continued)



NOTE

1 NOMENCLATURE IS ON BOTH TOP AND FRONT FRACE OF KEYBUTTON AS SHOWN. THE NUMBER TO THE UPPER LEFT DESIGNATES THE BUTTON POSITION.

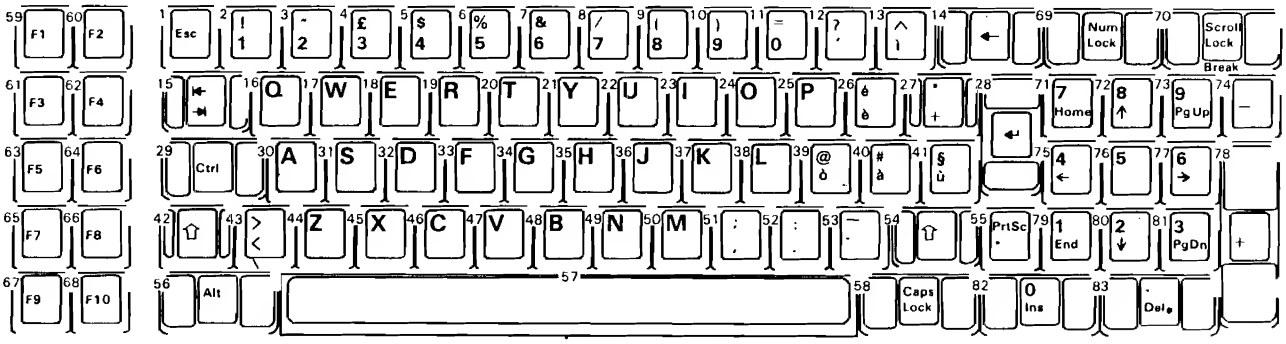
Figure 6. KEYBOARD DIAGRAM (Continued)



NOTE

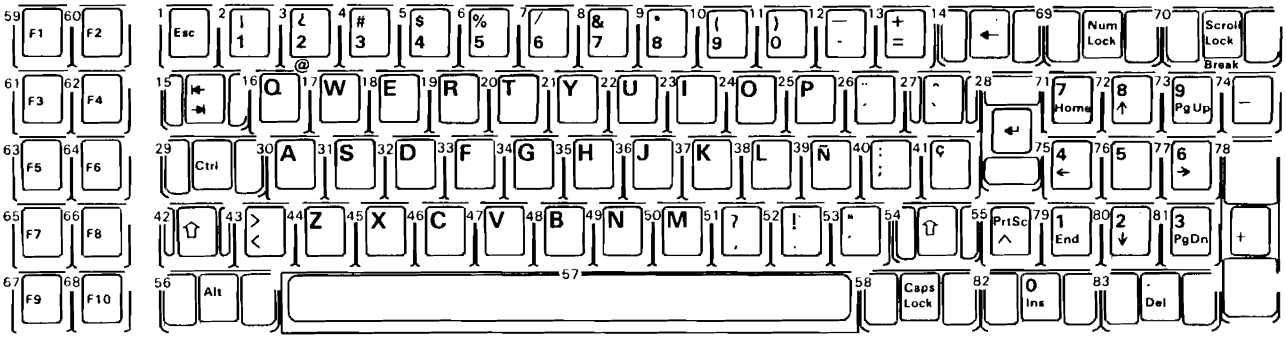
1 NOMENCLATURE IS ON BOTH TOP AND FRONT FACE OF KEYBUTTON AS SHOWN. THE NUMBER TO THE UPPER LEFT DESIGNATES THE BUTTON POSITION.

Figure 6 (Continued). KEYBOARD DIAGRAM



NOTE
1 NOMENCLATURE IS ON BOTH TOP AND FRONT FACE OF KEYBUTTON AS SHOWN. THE NUMBER TO THE UPPER LEFT DESIGNATES THE BUTTON POSITION.

Figure 6. KEYBOARD DIAGRAM (Continued)



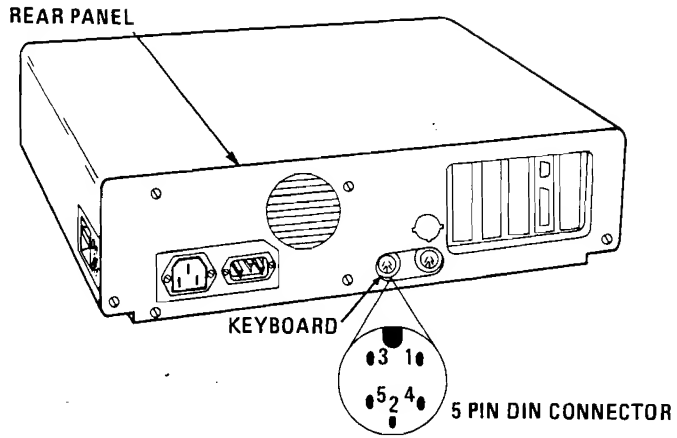
NOTE
1 NOMENCLATURE IS ON BOTH TOP AND FRONT FACE OF KEYBUTTON AS SHOWN. THE NUMBER TO THE UPPER LEFT DESIGNATES THE BUTTON POSITION.

Figure 6. KEYBOARD DIAGRAM (Continued)

Table 1. Keyboard Scan Codes

Key Position	Scan Code in Hex	Key Position	Scan Code in Hex
1	01	43	2B
2	02	44	2C
3	03	45	2D
4	04	46	2E
5	05	47	2F
6	06	48	30
7	07	49	31
8	08	50	32
9	09	51	33
10	0A	52	34
11	0B	53	35
12	0C	54	36
13	0D	55	37
14	0E	56	38
15	0F	57	39
16	10	58	3A
17	11	59	3B
18	12	60	3C
19	13	61	3D
20	14	62	3E
21	15	63	3F
22	16	64	40
23	17	65	41
24	18	66	42
25	19	67	43
26	1A	68	44
27	1B	69	45
28	1C	70	46
29	1D	71	47
30	1E	72	48
31	1F	73	49
32	20	74	4A
33	21	75	4B
34	22	76	4C
35	23	77	4D
36	24	78	4E
37	25	79	4F
38	26	80	50
39	27	81	51
40	28	82	52
41	29	83	53
42	2A		

Keyboard Interface Connector Specifications



PIN	SIGNAL
1	+ Keyboard Clock
2	+ Keyboard Data
3	- Keyboard Reset (Not used by keyboard)
4	Ground
5	+5 Volts

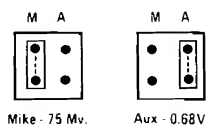
Cassette User Interface

The cassette interface control is implemented in software. (See FIRMWARE Section). An 8253 timer output is used to control the data to the cassette recorder. This output exits the System Board, at the rear, through pin 5 of a DIN connector. The cassette input data is read by an 8255A-5 Programmable Peripheral Interface (PPI) input port bit. This signal is received through pin 4 of the cassette connector. Software algorithms are used to generate and read cassette data. The cassette drive motor is controlled through pins 1 & 3 of the cassette connector. The motor on/off is controlled by an 8255A-PPI output port bit. (Port '61H', bit 3). The 8255A address and bit assignments are defined in the I/O Address Map page. On the following pages are read, write, and motor control block diagrams.

Cassette Jumpers

A 2x2 Berg Pin and Jumper are used on the cassette Data Out line. The jumper will allow the Data Out line to be used as a microphone input (75 mv.) when the jumper is placed across M and C pins. An auxiliary input is available when the jumper is placed across the A and C pins. The auxiliary input provides a .68 volt input to the recorder. Refer to System Board Component Diagram page (2-13) for cassette jumper location.

JUMPER DIAGRAM



Circuit Block Diagrams

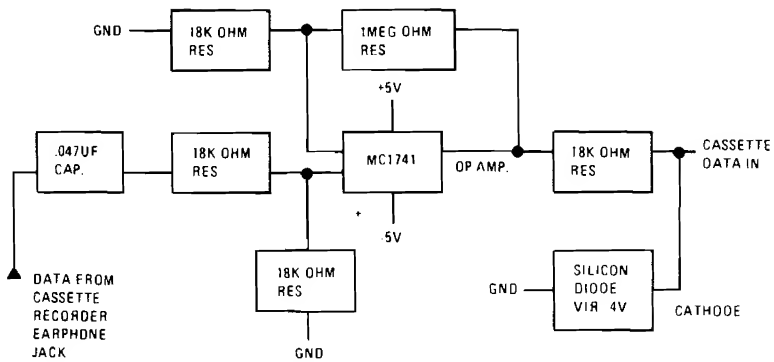


Figure 7. CASSETTE INTERFACE READ HARDWARE

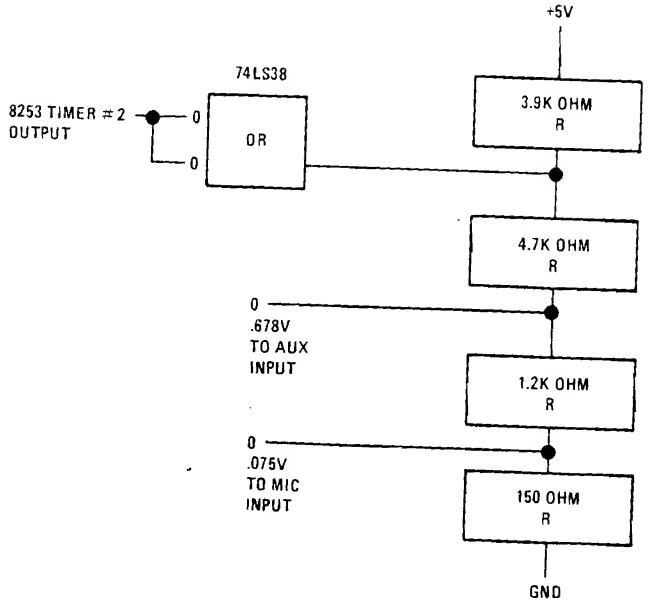


Figure 8. CASSETTE INTERFACE WRITE HARDWARE

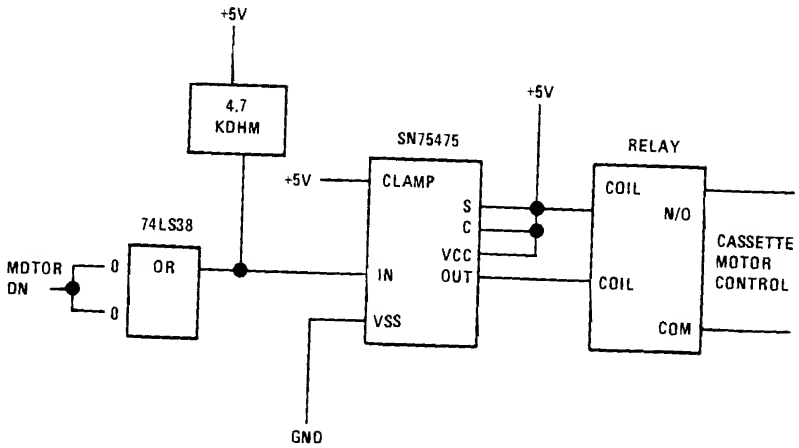
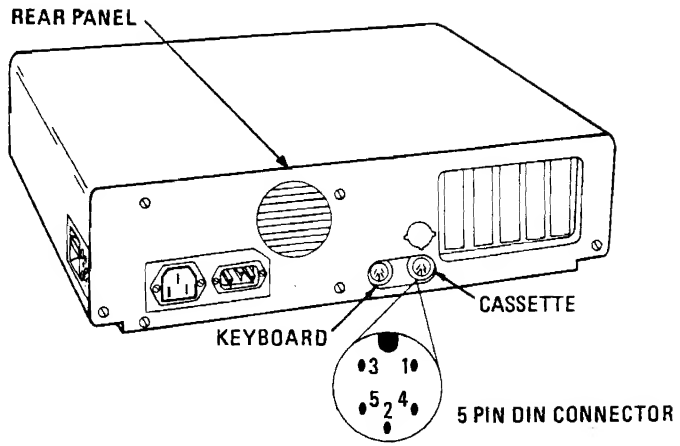


Figure 9. CASSETTE MOTOR CONTROL

Cassette Interface Connector Specifications



PIN	SIGNAL	ELECTRICAL CHARACTERISTICS*
1	Motor Control	Common from Relay
2	Ground	
3	Motor Control	6 VDC; 1A (Relay N.O.)
4	Data In	500nA at ±13V - at 1,000 - 2,000 Baud
5	Data Out (Mic or Aux)	250 μA at .68V or 75mv **

*All voltages and currents are maximum ratings and should not be exceeded.

**Data out can be chosen using a jumper located on planar. (AUX → .68V or MIC → 75 mV).

Interchange of these voltages on the cassette recorder could lead to damage of recorder inputs.

Speaker Interface

The sound system contains a small permanent magnet 2-1/4" speaker. The speaker can be driven from one or both of two sources. The sources are:

1. An 8255A-5 PPI output bit. The address and bit are defined in the I/O Address Map pages 2-23 and 2-24.
2. A timer Channel Clock out where the output is programmable within the functions of the 8253-5 timer with a 1.19 Mhz clock input. The timer gate is also controlled by an 8255A-5 PPI output port bit. Address and bit assignment are in the I/O Address Map pages 2-23 and 2-24.

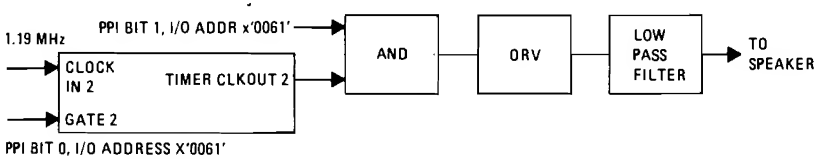


Figure 10. SPEAKER DRIVE SYSTEM BLOCK DIAGRAM

Channel 2 (Tone generation for Speaker)
- GATE 2 – Controlled by 8255A-5 PPI Bit (See I/O Map)
- CLK IN 2 – 1.19318 Mhz OSC
- CLK OUT 2 – Used to drive Speaker
  – Used to write data on the Audio Cassette

Speaker Connection - 4 Pin Berg Connector, Refer to System Board Diagram page 2-13 for speaker connection.

PIN	FUNCTION
1	DATA
2	KEY
3	GROUND
4	+5 VOLTS

I/O Address Map

HEX RANGE	9	8	7	6	5	4	3	2	1	0	DEVICE
00-0F	0	0	0	0	0	Z	A3	A2	A1	A0	DMA CHIP 8237-2
20-21	0	0	0	0	1	Z	Z	Z	Z	A0	INTERRUPT 8259A
40-43	0	0	0	1	0	Z	Z	Z	A1	A0	TIMER 8253-5
60-63	0	0	0	1	1	Z	Z	Z	A1	A0	PPI 8255A-5
80-83	0	0	1	0	0	Z	Z	Z	A1	A0	DMA PAGE REGS
*AX	0	0	1	0	1						NMI MASK REG
CX	0	0	1	1	0						RESERVED
EX	0	0	1	1	1						RESERVED
200-20F	0	0	0	0	0	0	A3	A2	A1	A0	GAME I/O ADAPTER
278-27F	1	0	0	1	1	1	1	Z	A1	A0	RESERVED
2F8-2FF	1	0	1	1	1	1	1	A2	A1	A0	ALT RS232 CARD
300-31F	1	1	0	0	0	A4	A3	A2	A1	A0	PROTOTYPE CARD
378-37F	1	1	0	1	1	1	1	Z	A1	A0	PARALLEL PRINTER PORT
280-38F	1	1	1	0	1	1	A3	A2	A1	A0	IBM MONOCHROME DISPLAY/ PARALLEL PRINTER ADAPTER
3D0-3DF	1	1	1	1	0	1	A3	A2	A1	A0	CDLOR/GRAPHICS ADAPTER
3F0-3F7	1	1	1	1	1	1	0	A2	A1	A0	5-1/4" DRIVE ADAPTER
3F8-3FF	1	1	1	1	1	1	1	A2	A1	A0	TP RS232 CARD

Z = Don't Care, i.e., Not in Decode

* At power on time, the Non Mask Interrupt NMI into the 8088 is masked off. This mask bit can be set and reset via system software as follows:

Set mask: write X'80' to I/O Address X'A0' (enable NMI)

Clear mask: write X'00' to I/O Address X'A0' (disable NMI)

I/O Address Map

X'0060'	INPUT	PA0	1	+KBD SCAN CODE	0	IPL 5 1/4 DISKETTE DRIVE	(SW1-1)
		1	2		1	RESERVED	(SW1-2)
		2	3		2	SYS. BD. READ/WRITE MEMORY SIZE *	(SW1-3)
		3	4		3	SYS. BD. READ/WRITE MEMORY SIZE *	(SW1-4)
		4	5	DR	4	+DISPLAY TYPE 1 **	(SW1-5)
		5	6		5	+DISPLAY TYPE 2 **	(SW1-6)
		6	7		6	NO. OF 5 1/4 DRVS ***	(SW1-7)
		7	8		7	NO. OF 5 1/4 DRVS ***	(SW1-8)
X'0061'	OUTPUT	PB0	1	+TIMER 2 GATE SPEAKER			
		0	2	+SPEAKER DATA			
		1	3	+(READ READ/WRITE MEMORY SIZE) OR (READ SPARE KEY)			
		2	4	+CASSETTE MOTOR OFF			
		3	5	-ENABLE READ/WRITE MEMORY			
		4	6	-ENABLE I/O CH CK			
		5	7	-HOLD KBD CLK LOW			
		6	8	-(ENABLE KBD) OR + (CLR KBD & ENABLE SENSE SW'S)			
		7					
X'0062'	INPUT	PC0	1	I/O READ/WRITE MEMORY (SW2-1)			
		1	2	I/O READ/WRITE MEMORY (SW2-2)	BINARY VALUE X 32KB	OR	SPARE KEY (SW2-5)
		2	3	I/O READ/WRITE MEMORY (SW2-3)			
		3	4	I/O READ/WRITE MEMORY (SW2-4)			
		4	5	+CASSETTE DATA IN			
		5	6	+TIMER CHANNEL 2 OUT			
		6	7	+I/O CHANNEL CHECK			
		7	8	+READ/WRITE MEMORY PCK			

X'0063' CMO/MODE REGISTER

MODE REG VALUE

7	6	5	4	3	2	1	0
1	0	0	1	1	0	0	1

X'99'

*

PA3 SW1-4	PA2 SW1-3	AMOUNT OF MEMORY LOCATED ON SYS. BD.
0	0	16K BYTES
0	1	32K BYTES
1	0	48K BYTES
1	1	64K BYTES

**

PA5 SW1-6	PA4 SW1-5	TYPE OF DISPLAY
0	0	RESERVED
0	1	COLOR CARD 40X25 (BW MODE)
1	0	COLOR CARD 80X25 (BW MODE)
1	1	IBM MONOCHROME DISPLAY (80 X 25)

***

PA7 SW1-8	PA6 SW1-7	NUMBER OF 5-1/4" DRIVES IN SYSTEM
0	0	1
0	1	2
1	0	3
1	1	4

8255A - I/O BIT MAP

NOTE: PA bit=0 implies switch "ON".
PA bit=1 implies switch "OFF".

System Memory Map

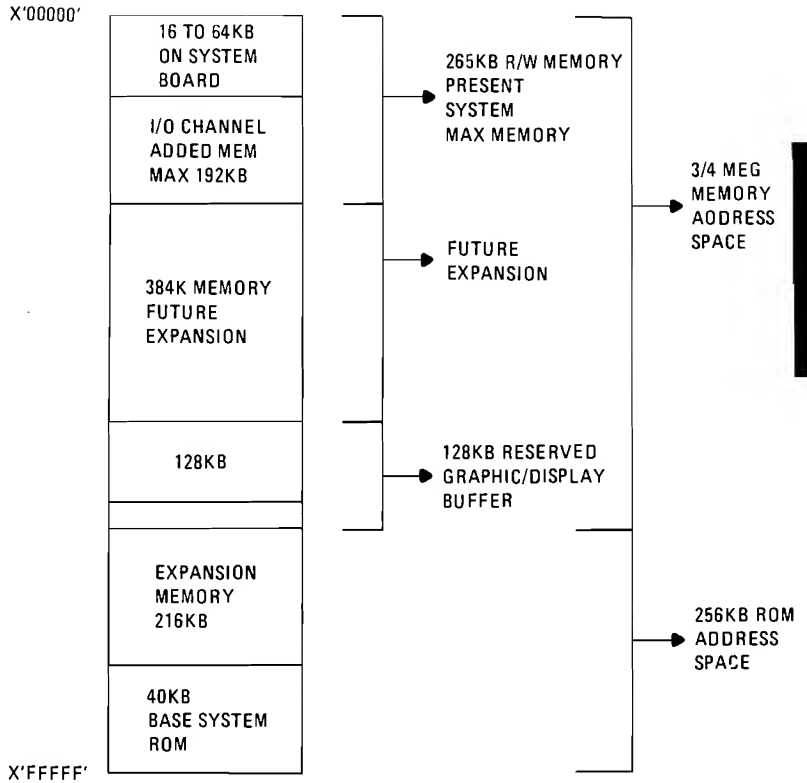


Figure 11. SYSTEM MEMORY MAP

System Memory Map (Increments of 16KB)

START ADDRESS: DECIMAL	HEX	FUNCTION:
0	00000	16–64 KB READ/WRITE MEMORY ON SYSTEM BOARD
16K	04000	
32K	08000	
48K	0C000	
64K	10000	UP TO 192 KB MEMORY IN I/O CHANNEL
80K	14000	
96K	18000	
112K	1C000	
128K	20000	
144K	24000	
160K	28000	
178K	2C000	
192K	30000	
208K	34000	
224K	38000	
240K	3C000	
256K	40000	384 KB FUTURE R/W MEMORY EXPANSION IN I/O CHANNEL
272K	44000	
288K	48000	
304K	4C000	
320K	50000	
336K	54000	
352K	58000	
368K	5C000	
384K	60000	
400K	64000	
416K	68000	
432K	6C000	
448K	70000	
464K	74000	
480K	78000	
496K	7C000	
512K	80000	
528K	84000	
544K	88000	
560K	8C000	
576K	90000	
592K	94000	
608K	98000	
624K	9C000	

Figure 12. SYSTEM MEMORY MAP (INCREMENTS OF 16KB) (SHEET 1 OF 2)

System Memory Map Cont.

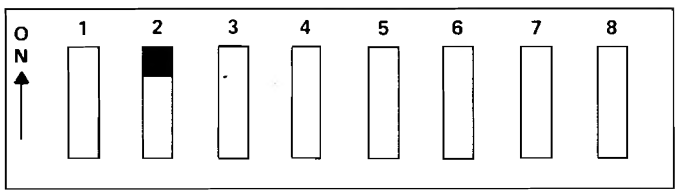
START ADDRESS: DECIMAL	HEX	FUNCTION:
640K	A0000	RESERVED
656K	A4000	112 KB GRAPHICS/DISPLAY VIDEO BUFFER
672K	A8000	
688K	AC000	
704K	B0000	MONOCHROME
720K	B4000	
736K	B8000	COLOR/GRAPHICS
752K	BC000	
768K	C0000	192 KB MEMORY EXPANSION AREA
784K	C4000	
800K	C8000	
816K	CC000	
832K	D0000	
848K	D4000	
864K	D8000	
880K	DC000	
896K	E0000	
912K	E4000	
928K	E8000	
944K	EC000	
960K	F0000	RESERVED
976K	F4000	48 KB BASE SYSTEM ROM
992K	F8000	
1.008M	FC000	

Figure 12. SYSTEM MEMORY MAP (16KB) (SHEET 2)

System Board and Memory Expansion Switch Settings

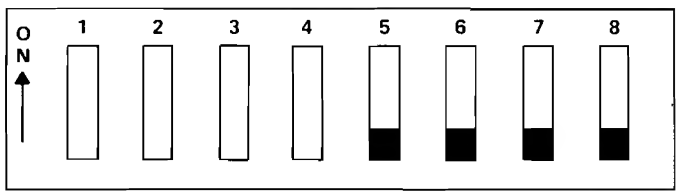
On the following four pages are graphic illustrations of switch settings. These are necessary for the system to address components attached, and to specify the amount of memory installed both on the System Board and in the System Expansion Slots. Refer to the System Board Component Diagram (page 2-13) for DIP switch locations.

SWITCH 1



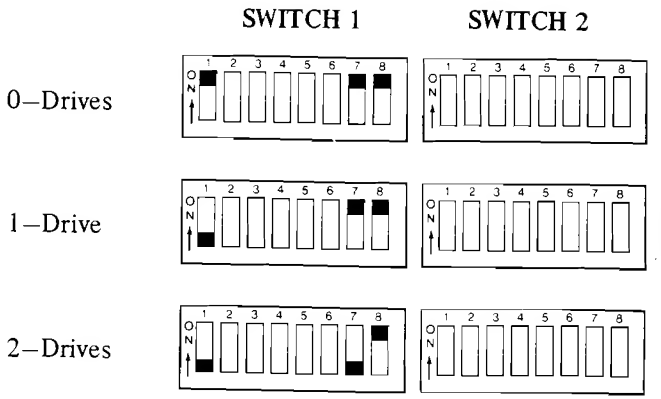
Position	Function
1-7-8	Number of 5-1/4" Diskette Drives Installed; page 2-29
2	Unused – must be on (reserved for co-processor)
3-4	Amount of memory on System Board; pages 2-30 to 2-31
5-6	Type of monitor you are using; page 2-29

SWITCH 2

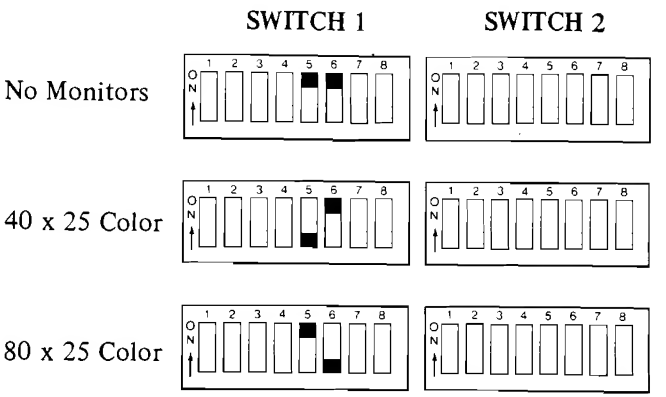


Position	Function
1-2-3-4	Amount of memory options installed; page 2-30 to 2-31
5-6-7-8	Always in the OFF position

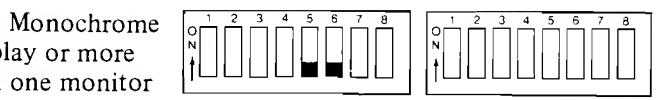
Number of 5-1/4" Diskette Drives Installed



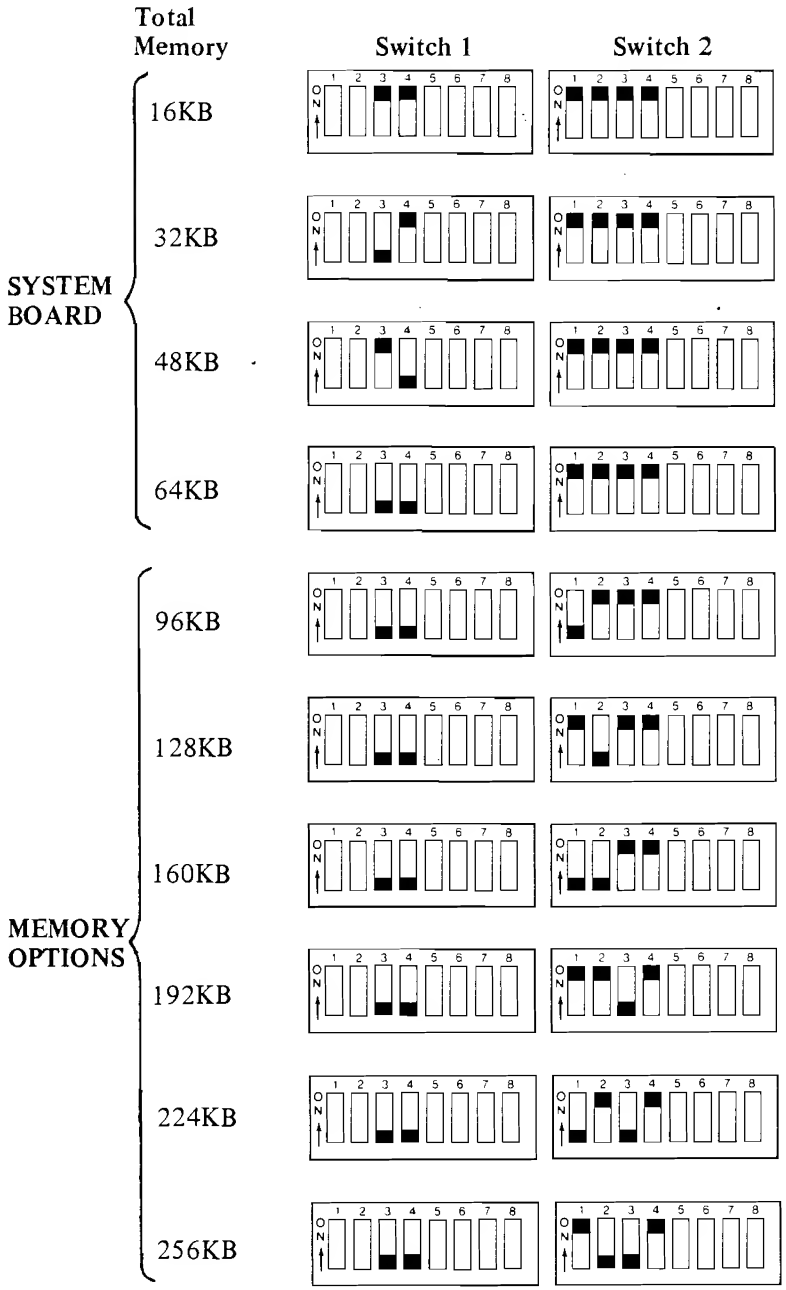
Monitor Type



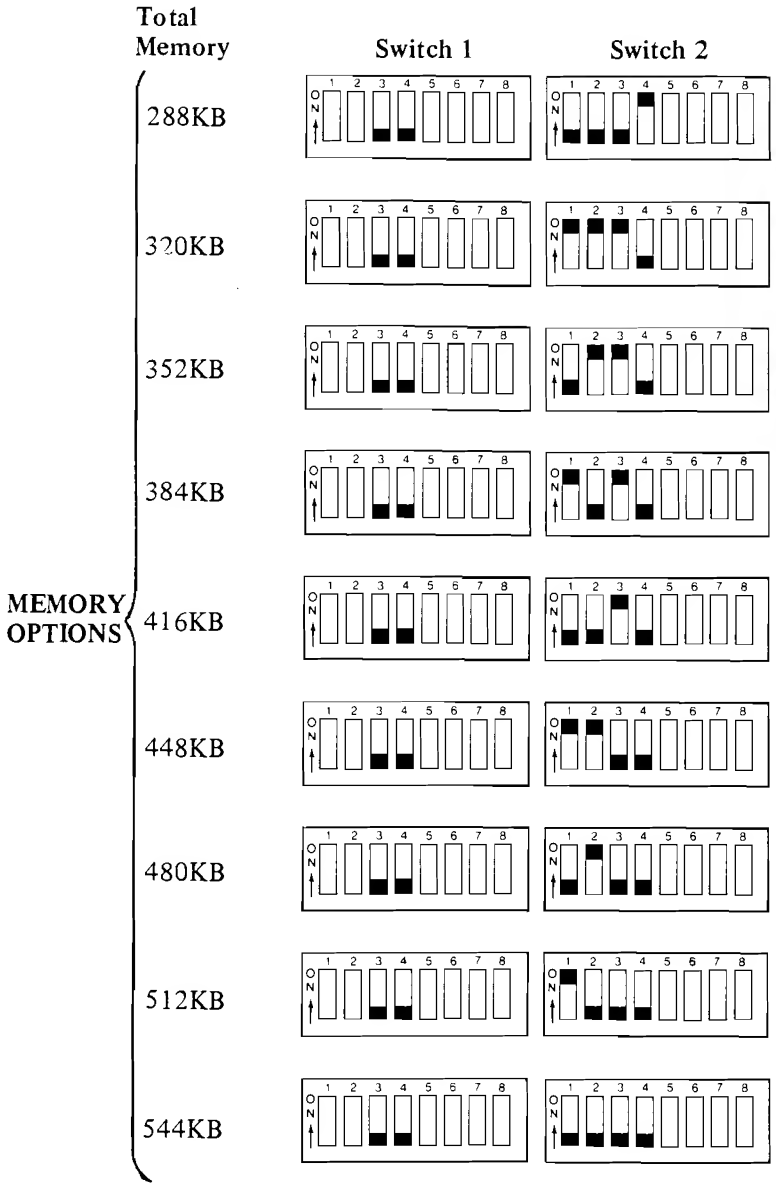
Note: The 80 x 25 color setting, when used with home television and various monitors can cause loss of character/quality.



System Board Memory Switch Settings

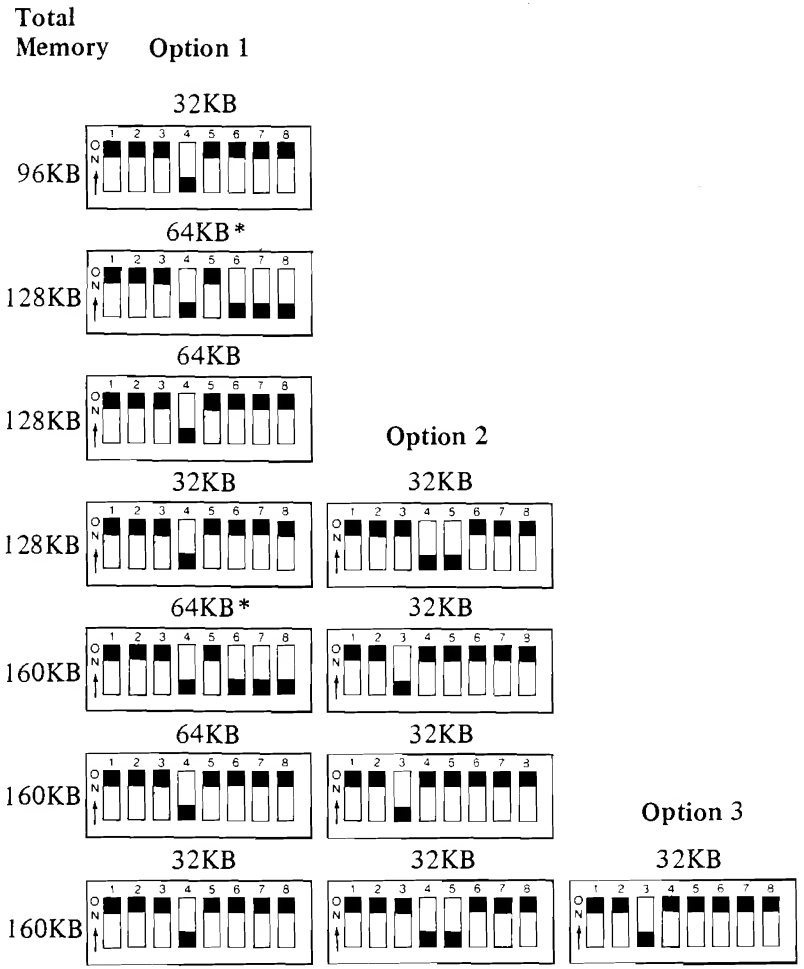


System Board Memory Switch Settings



Memory Expansion Options

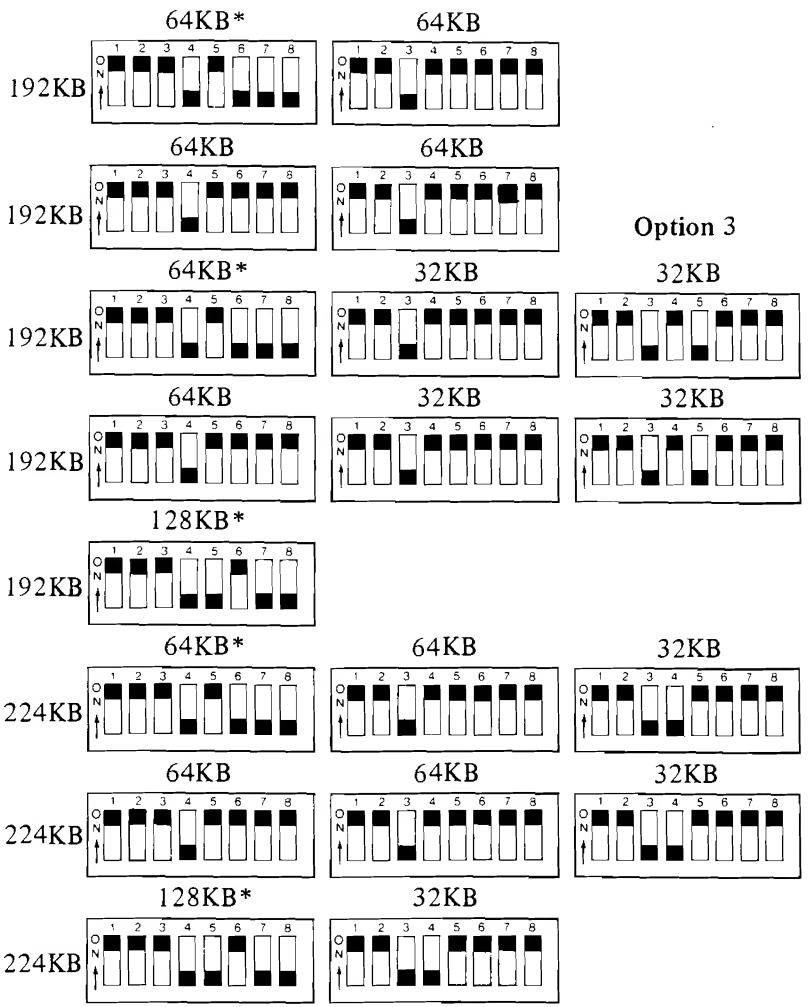
The following are switch settings on each Memory Expansion Option.



*Amount of memory installed on a 64/256KB Memory Expansion Option.

Memory Expansion Options

The following are switch settings on each Memory Expansion Option.

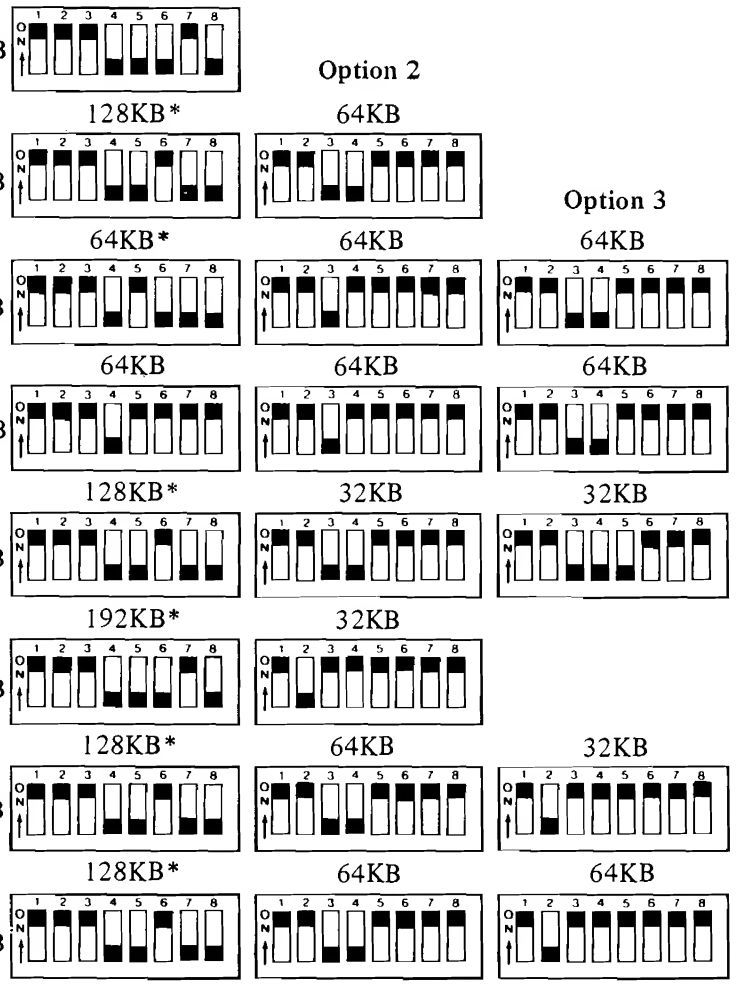


*Amount of memory installed on a 64/256KB Memory Expansion Option.

Memory Expansion Options

The following are switch settings on each Memory Expansion Option.

Total Memory	Option 1	Option 2	Option 3
256KB	192KB*		
256KB	128KB*	64KB	
256KB	64KB*	64KB	64KB
256KB	64KB	64KB	64KB
256KB	128KB*	32KB	32KB
288KB	192KB*	32KB	
288KB	128KB*	64KB	32KB
320KB	128KB*	64KB	64KB

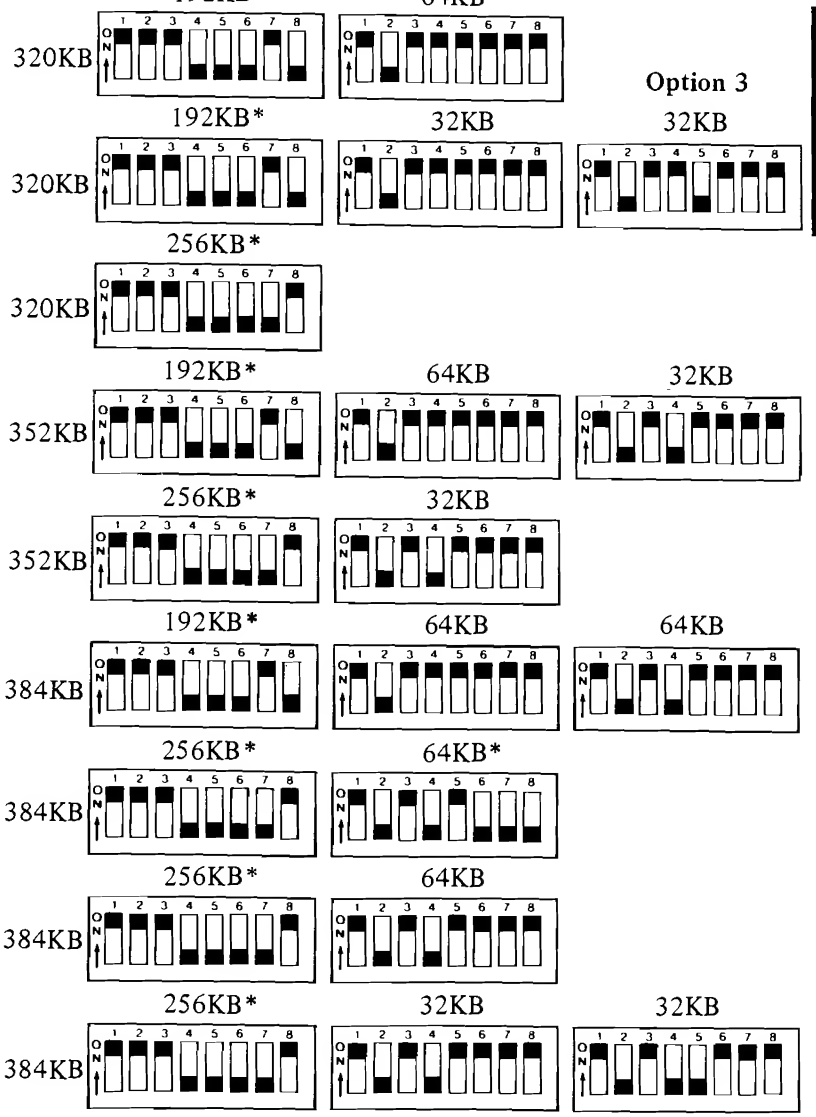


*Amount of memory installed on a 64/256KB Memory Expansion Option.

Memory Expansion Options

The following are switch settings on each Memory Expansion Option.

Total Memory	Option 1	Option 2	Option 3
320KB	192KB*	64KB	
320KB	192KB*	32KB	32KB
320KB	256KB*		
352KB	192KB*	64KB	32KB
352KB	256KB*	32KB	
384KB	192KB*	64KB	64KB
384KB	256KB*	64KB*	
384KB	256KB*	64KB	
384KB	256KB*	32KB	32KB

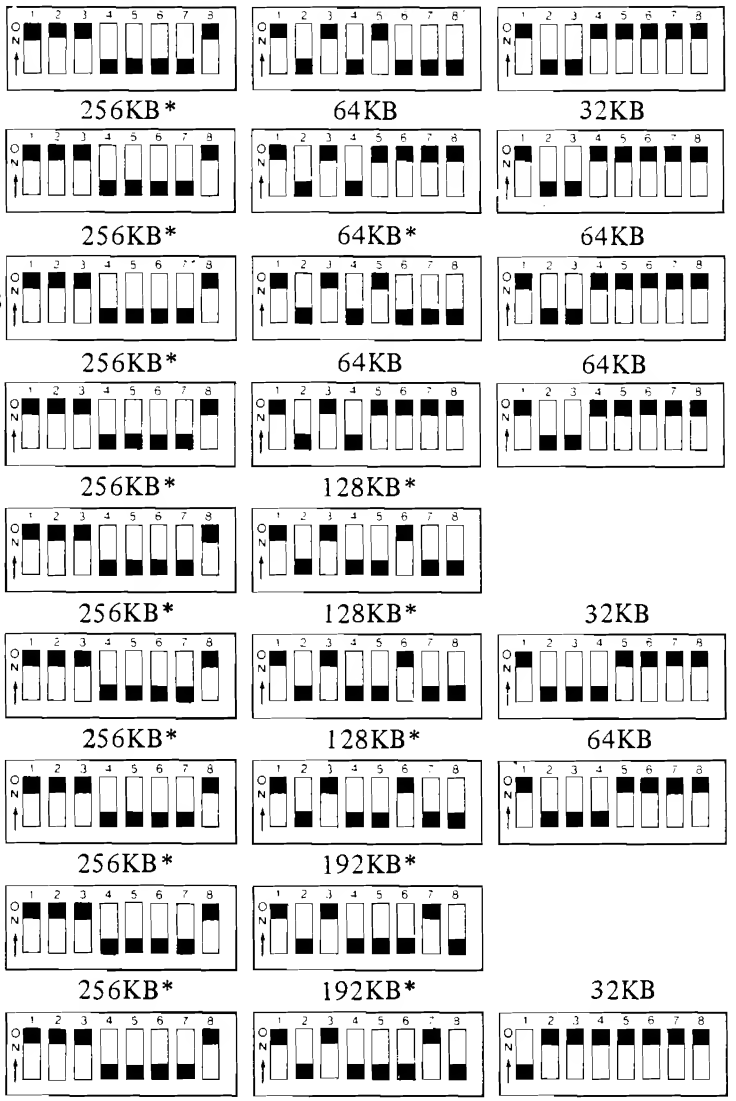


*Amount of memory installed on a 64/256KB Memory Expansion Option.

Memory Expansion Options

The following are switch settings on each Memory Expansion Option.

Total Memory	Option 1	Option 2	Option 3
416KB	256KB*	64KB*	32KB
416KB	256KB*	64KB	32KB
448KB	256KB*	64KB*	64KB
448KB	256KB*	64KB	64KB
448KB	256KB*	128KB*	
480KB	256KB*	128KB*	32KB
512KB	256KB*	128KB*	64KB
512KB	256KB*	192KB*	
544KB	256KB*	192KB*	32KB



*Amount of memory installed on a 64/256KB Memory Expansion Option.

Power Supply

The system DC power supply is a 63.5 watt, 4 voltage level switching regulator. It is integrated into the System Unit and supplies power for the System Unit, its options, and the keyboard. The supply provides 7 amps of +5 Vdc, ±5% 2 amps of +12 Vdc, ±5% 300 ma of -5 Vdc, ±10% and 250 ma of -12 Vdc, ±10%. All power levels are regulated with overvoltage and over current protection. If DC over-load or over-voltage conditions exist, the supply will automatically shut down until the condition is corrected. The supply is designed for continuous operation at 63.5 watts and the input to the supply is fused.

There are two different power supplies available for the system unit: one that has an input voltage of 120 Vac and one that has an input voltage of 220/240 Vac. Other than their input ratings the two power supplies have identical specifications.

The System Board takes approximately 3 amps of +5 Vdc thus allowing approximately 4 amps of 5 Vdc for the adapters in the System Expansion Slots. The +12 Vdc power level is designed to power the two internal 5-1/4" Diskette Drives and the system's dynamic memory. It is assumed that only one drive motor is active at a time. The -5 Vdc level is used for memory bias voltage and analog circuits in the diskette adapter phase lock loop. The +12 Vdc and -12 Vdc are used for powering the serial interface card EIA drivers and receivers for the Asynchronous Communications Adapter. All four power levels are bussed across the five System Expansion Slots and available for option adapter.

The IBM Monochrome Display has its own power supply. This high resolution display receives its AC power from the System Unit power supply and is switched on and off with the power switch, which saves a wall outlet. The AC output for the display is a nonstandard connector, so only the AC high resolution Display can use this AC port.

Power Supply Location

The Power Supply is located at the right rear area of the System Unit. It supplies operating voltages to the System Board, IBM Monochrome Display, and provides two separate connections for power to the 5-1/4" Diskette Drives (if installed). The nominal power requirements and output voltages are listed on the following tables:

Input Requirements 120 Volt

VOLTAGE Vac			FREQUENCY Hz	CURRENT MAX
NOMINAL	MINIMUM	MAXIMUM	± 3 Hz	@Vin MINIMUM
120	104	127	60	2.5 AMPS @ 104 Vac

Input Requirements 220/240 Volt

VOLTAGE Vac			FREQUENCY Hz	CURRENT MAX
NOMINAL	MINIMUM	MAXIMUM	± 3 Hz	@Vin MINIMUM
220/240	180	259	50	1.0 AMPS @ 180 Vac

DC Output

VOLTAGE Vdc	CURRENT AMPS		REGULATION TOLERANCE	
NOMINAL	MIN	MAX	±%	−%
+ 5.0	2.3	7.0	5	4
− 5.0	0.0	0.3	10	8
+12.0	0.0	2.0	5	4
−12.0	0.0	0.25	10	9

AC Output

The AC voltage out is connected to the input voltage through the power supply fuse filter card and On/Off switch. Since this output is designed to support only the IBM Monochrome Display and since the supply input is fused, there is only a limited amount of current available: .75 A maximum on the 120 Vac supply and .38 A maximum on the 220/240 Vac power supply.

Power Supply Connectors and Pin Assignment

The power connector on the System Board is a 12 pin male connector which plugs into the power supply connectors. The pin configurations and locations are shown below:

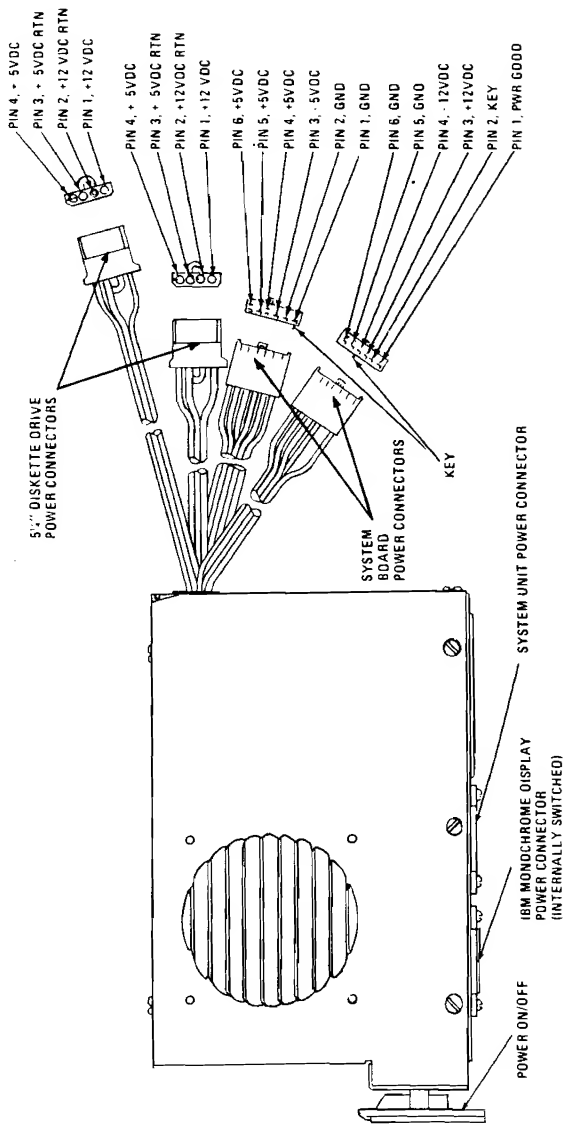


Figure 13. POWER SUPPLY AND CONNECTORS

Important Operating Characteristics

Over Voltage/Current Protection

PRIMARY (INPUT)

VOLTAGE NOMINAL VAC	TYPE PROTECTION	RATING AMPS
120	60 Hz FUSE TYPE 2 SOC SD4	2 AMPS
240	50 Hz FUSE TYPE 2 SOC SD4	1 AMP

Power On/Off Cycle: When the supply is turned off for a minimum of 5 seconds, and then turned on, the power good signal will be regenerated.

Signal Requirements

The power good signal indicates that there is adequate power to continue processing. If the power goes below the specified levels, the power good signal triggers a system shut-down.

The Power Supply provides a power good signal out, to indicate that the +/-5V and +/-12V outputs are above the sense level defined in the chart below. The power good signal is up level (2.4V to 5.5V), TTL compatible, and capable of sourcing 60 UA. When any of the four sensed output voltages is below its sense level voltage as defined in the chart below, the power good signal is down level (0V to 0.4V), TTL compatible, and capable of sinking 500 UA. The power good signal (after all levels of the output voltage are good) has a turn on delay of at least 100 MS but no more than 500 MS.

The sense levels of the +/-5V and +/-12V outputs are:

OUTPUT	MIN	SENSE VOLTAGE NOMINAL	MAX
+5V	+3.7	+4.0	+4.3
-5V	-3.7	-4.0	-4.3
+12V	+8.5	+9.6	+10.5
-12V	-8.5	-9.6	-10.5

IBM Monochrome Display and Parallel Printer Adapter

This adapter has dual functions. The first is to provide the interface to the IBM Monochrome Display. The second function is a parallel interface for the IBM 80 CPS Matrix Printer.

The monitor interface is designed around the Motorola 6845 CRT Controller module. There are 4K bytes of static memory on the card which are used for the display buffer. The memory is dual ported and may be accessed directly by the CPU. No parity is provided on the display buffer. A block diagram of the Monochrome Display function in on page 2-48.

The characteristics of the design are listed below:

- 80x25 Screen
- Direct Drive Output
- 9x14 Character Box
- 7x9 Character
- 18 Khz Monitor
- Character Attributes

The adapter supports 256 character codes. An 8K byte character generator contains the fonts for the character codes. The characters, values, keystrokes and screen characteristics are tabled in Appendix C. Of Characters, Keystrokes and Color.

Note: This Adapter when used with a display containing P39 Phospor, will not support a light pen!

Parallel Interface Description

This topic is discussed in full on pages 2-75 through 2-79.

IBM Monochrome Display Adapter Block Diagram

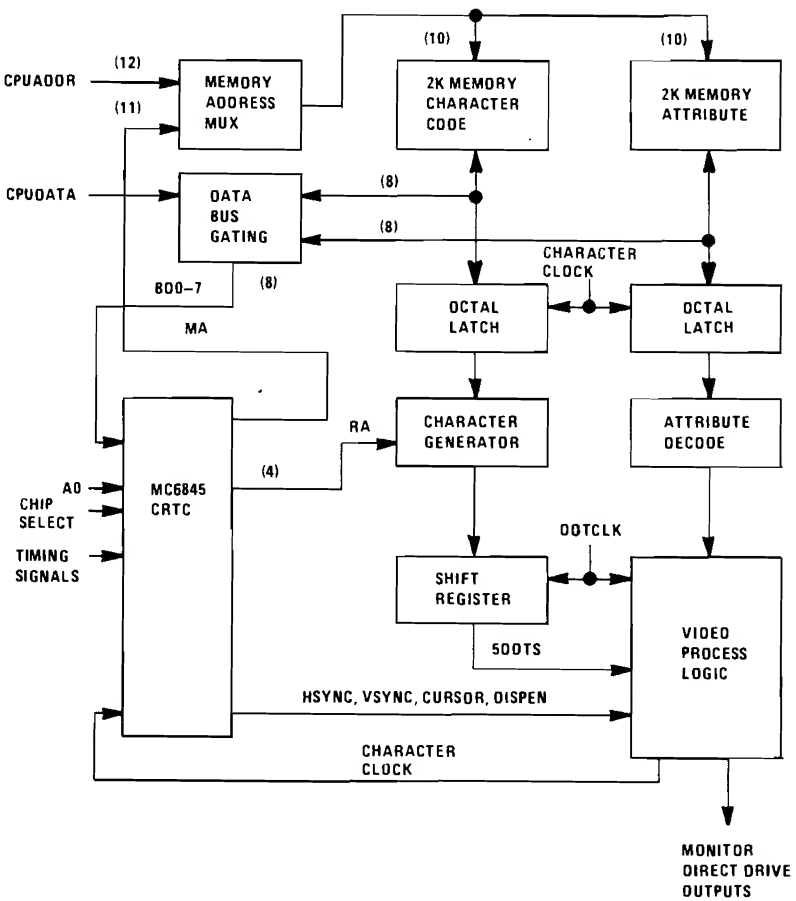


Figure 14. IBM MONOCHROME DISPLAY ADAPTER BLOCK DIAGRAM

System Channel Interface

Lines Used

This card uses the address and data bus, memory and I/O read/write signals, reset, I/O Ready, I/O Clock, and IRQ7.

Loads

Where possible, only one "LS" load is on the signals present at the I/O slot. Some of the address bus lines have two "LS" loads. No signal has more than two "LS" loads.

Special Timing

At least one wait state will be inserted on all memory and I/O accesses from the CPU. The duration of the wait-state will vary because the CPU/monitor access is synchronized with the character clock on this adapter.

To insure proper initialization of the attachment, the first instruction issued to the card must be to set the high resolution bit of the monitor output Port 1. (OUT PORT 3B8 = 01H). A CPU access to this adapter must never occur if the high resolution bit is not set.

System configurations which have two display adapter cards must insure that both adapters are properly initialized after a power on reset. Damage to either display may occur if not properly initialized.

Data Rates

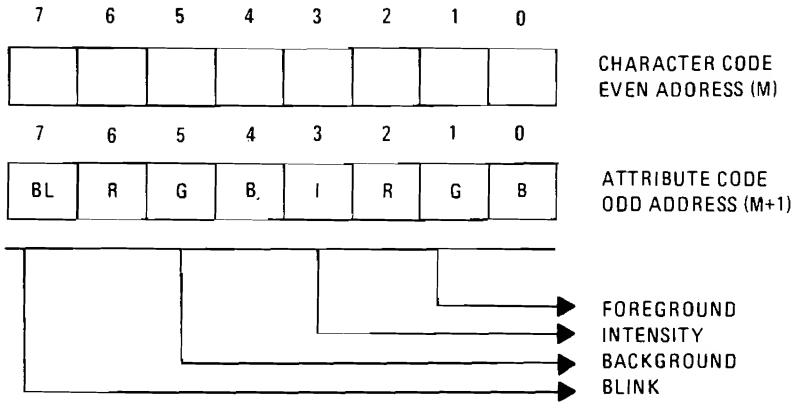
For the IBM Monochrome Display Adapter, two bytes are fetched from the display buffer in 553 ns providing a data rate of 1.8M bytes/second.

Interrupt and DMA Response Requirements

- The display buffer can be written into, or read from using DMA.
- The parallel interface uses the +IRQ7 line. Interrupt becomes active when the acknowledge input is low, and interrupts are enabled via the control port.

Modes of Operation

The IBM Monochrome Display and Printer Adapter supports 256 character codes. In the character set are alphanumerics and block graphics. Each character in the display buffer has a corresponding character attribute. The character code must be an even address and the attribute code must be an odd address in the display buffer.



The adapter decodes the character attribute byte as defined above. The BLINK and INTENSITY bits may be combined with the FOREGROUND and BACKGROUND bits to further enhance the character attribute functions listed below.

BACKGROUND R G B	FOREGROUND R G B	FUNCTION
0 0 0	0 0 0	NON DISPLAY
0 0 0	0 0 1	UNDERLINE
0 0 0	1 1 1	WHITE CHARACTER/ BLACK BACKGROUND
1 1 1	0 0 0	REVERSE VIDEO

Programming Considerations

Programming the 6845 CRT Controller

The following table summarizes the 6845 Internal Data Registers and their functions and parameters. For the IBM Monochrome Display, the values in the table must be programmed into the 6845 to insure proper initialization of the device.

Table 2. 6845 Initialization Parameters

REGISTER #	REGISTER FILE	PROGRAM UNIT	80x25 MONOCHROME
R0	HORIZONTAL TOTAL	CHARACTERS	61H
R1	HORIZONTAL DISPLAYED	CHARACTERS	50H
R2	HSYNC POSITION	CHARACTERS	52H
R3	HSYNC WIDTH	CHARACTERS	FH
R4	VERTICAL TOTAL	CHAR ROWS	19H
R5	VTOTAL ADJUST	SCAN LINE	6H
R6	VERTICAL DISPLAYED	CHAR ROW	19H
R7	VSYNC POSITION	CHAR ROW	19H
R8	INTERLACE MODE	---	02
R9	MAX SCAN LINE ADDRESS	SCAN LINE	DH
R10	CURSOR START	SCAN LINE	BH
R11	CURSOR END	SCAN LINE	CH
R12	START ADDRESS (H)	---	00H
R13	START ADDRESS (L)	---	00H
R14	CURSOR (H)	---	00H
R15	CURSOR (L)	---	00H
R16	RESERVED	---	---
R17	RESERVED	---	---

Sequence of Events

The first command issued to this attachment must be to output to PORT 3B8, hex 01, to set high resolution mode. If the high resolution mode is not set, an infinite CPU wait-state will occur!

Memory Requirements

The attachment has 4K bytes of memory which is used for the display buffer. The memory supports one screen of 25 rows of 80 characters, plus a character attribute for each display character. No parity is provided on the memory. No system Read/Write memory is required for the monochrome adapter portion. The display buffer starts at address 'B0000'.

DMA Channels

The display buffer will support a DMA operation, however CPU wait-states will be inserted during DMA.

Interrupt Levels

Interrupt Level 7 is used on the parallel interface. Interrupts can be enabled or disabled via the Printer Control Port. The interrupt is a high level active signal.

I/O Address and Bit Map

The table below breaks down the functions of the I/O Address decode for the card. The I/O address decode is from '3B0' through '3BF'. The bit assignment for each I/O address follows:

I/O Address	Function
3B0	Not Used
3B1	Not Used
3B2	Not Used
3B3	Not Used
3B4	6845 Index Register
3B5	6845 Data Register
3B6	Not Used
3B7	Not Used
3B8	CRT Control Port 1
3B9	Reserved
3BA	CRT Status Port
3BB	Reserved
3BC	Parallel Data Port
3BD	Printer Status Port
3BE	Printer Control Port
3BF	Not Used

The 6845 Index and Data Registers are used to program the CRT controller to interface to the high resolution Monochrome Display.

- CRT Output Port 1 (I/O Address '3B8')

Bit #	Function
0	+ high resolution mode
1	Not Used
2	Not Used
3	+ video enable
4	Not Used
5	+ enable blink
6, 7	Not Used

- CRT Status Port (I/O Address '3BA')

Bit #	Function
0	+ horizontal Drive
1	Reserved
2	Reserved
3	+ B/W Video

IBM Monochrome Display

The high resolution IBM Monochrome Display unit attaches to the System Unit via two cables of approximately 3' (914 mm) in length. One cable is a signal cable which contains direct drive interface from the IBM Monochrome Display and Printer Adapter.

The second cable provides AC power to the display from the System Unit. This allows the System Unit power On/Off switch to also control the display unit. An additional benefit is a reduction in the requirements for wall outlets to power the system. The monitor contains an 11.5" (283.1 mm) diagonal 90° deflection CRT. The CRT and analog circuits are packaged in an enclosure so the display may either sit on top of the System Unit or on a nearby table top or desk. The unit has both brightness and contrast adjustment controls on the front available to the operator.

Operating Characteristics

Screen

High persistence green phosphor (P 39) with an etched surface to reduce glare. Unit displays an 80 character by 25 line screen with a 9 dot wide by 14 dot tall character box.

Video Signal

Maximum video bandwidth of 16.27 Mhz.

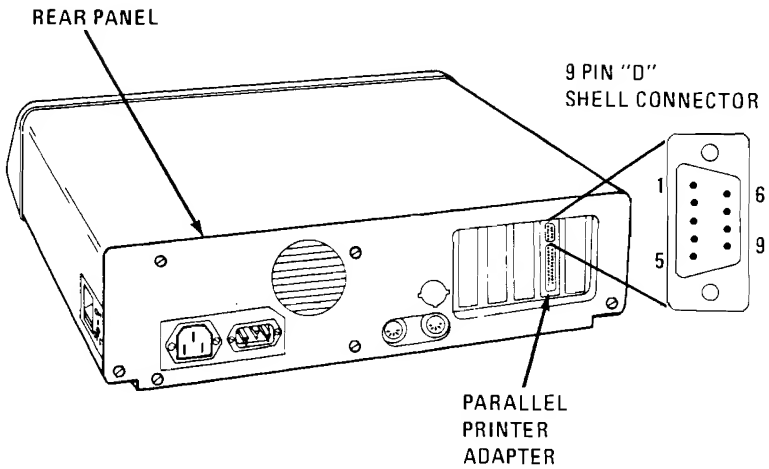
Vertical Drive

Screen refreshed at 50 Hz with 350 vertical lines of resolution and 720 lines of horizontal resolution measured at center of screen.

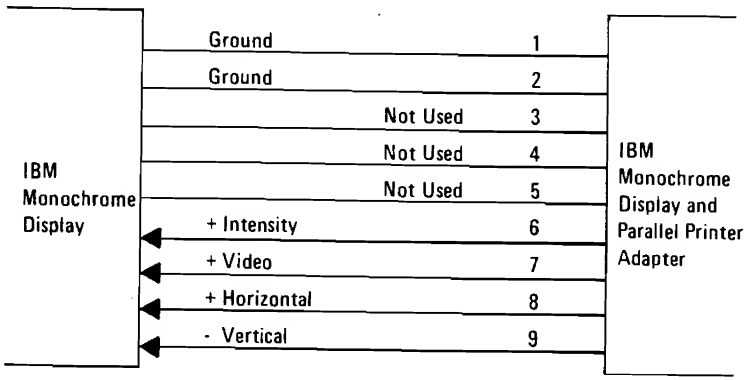
Horizontal Drive

Positive level TTL compatible frequency, 18.432 Khz.

IBM Monochrome Direct Drive Interface and Pin Assignment



At Standard TTL Levels



NOTE: Signal voltages are 0 - .6 Vdc at down level
+5 Vdc at high level

Color/Graphics Monitor Adapter

The Color/Graphics Monitor Adapter is designed to attach a wide variety of NTSC (U.S. National Television System Committee) standard TV frequency monitors and TV sets (user-supplied RF modulator required for TVs). It is capable of operating in black and white or color, and provides three video interfaces: a composite video port, a direct drive port, and a connection interface for driving a user supplied RF modulator. In addition, a light pen interface is provided.

The adapter has two basic modes of operation; alphanumeric (A/N) and all points addressable graphics (APA). Additional modes are available within A/N and APA modes. In A/N mode, the display can be operated in a 40x25 mode for low resolution monitors and TVs or 80x25 mode for high resolution monitors. In both modes, characters are defined in an 8x8 box and are 7x7 with one line of descender for lowercase (both uppercase and lowercase characters are supported in all modes). In black and white mode, the character attributes of Reverse Video, Blinking and Highlighting are available. In color mode, there are 16 foreground colors and 8 background colors available per character. In addition, blinking on a per character basis is available.

The adapter card contains 16KB of storage; thus, for a 40x25 screen, 1000 bytes are used to store character information and 1000 bytes are used for attribute/color information. This means that up to 8 pages of screens can be stored in the adapter memory. Similarly, in an 80x25 mode, 4 pages of display screen may be stored in the adapter. The full 16KB storage on the display adapter is directly addressable by the processor allowing maximum software flexibility in managing the screen. In A/N color modes, it is also possible to select the screen border color. One of 16 colors may be selected.

In APA mode, there are two resolutions available; 320x200 and 640x200. In the 320x200, each (picture element) pel may have one of four colors. The background color (color 0) may be any of the 16 possible colors. The remaining 3 colors come from one of the two software selectable palettes. One palette contains red/green/brown, the other contains cyan/magenta/white.

The 640x200 mode is only available in black and white since the full 16KB of storage is used to define the on or off state of the pel.

The adapter operates in noninterlace mode at either 7 or 14 megahertz (Mhz) video bandwidth depending on the mode of operation selected. Vertical scan output frequency is 60 Hz and horizontal output frequency is 15,750 Hz.

In A/N mode, characters are formed from a ROM character generator. The character generator contains dot patterns for 256 characters. The character set contains the following major grouping of characters. Sixteen special characters for game support, 15 characters for support of word processing editing functions, the standard 96 ASCII graphic set, 48 characters to support foreign languages, 48 characters for business block graphics allowing drawing of charts, boxes and tables using single and double lines, 16 of the most often used Greek characters, and 15 of the most often used scientific notation characters.

The Color/Graphics Monitor Adapter Function is packaged on a single card which fits into one of the five System Expansions Slots on the System Board. The direct drive and composite video ports are right-angle mounted connectors at the rear of the adapter and extend through the rear panel of the System Unit.

The display adapter is implemented using a Motorola 6845 CRT controller device. This adapter is highly programmable with respect to raster and character parameters. A block diagram of the Color/Graphics Adapter is on the following page.

Color/Graphics Monitor Adapter Block Diagram

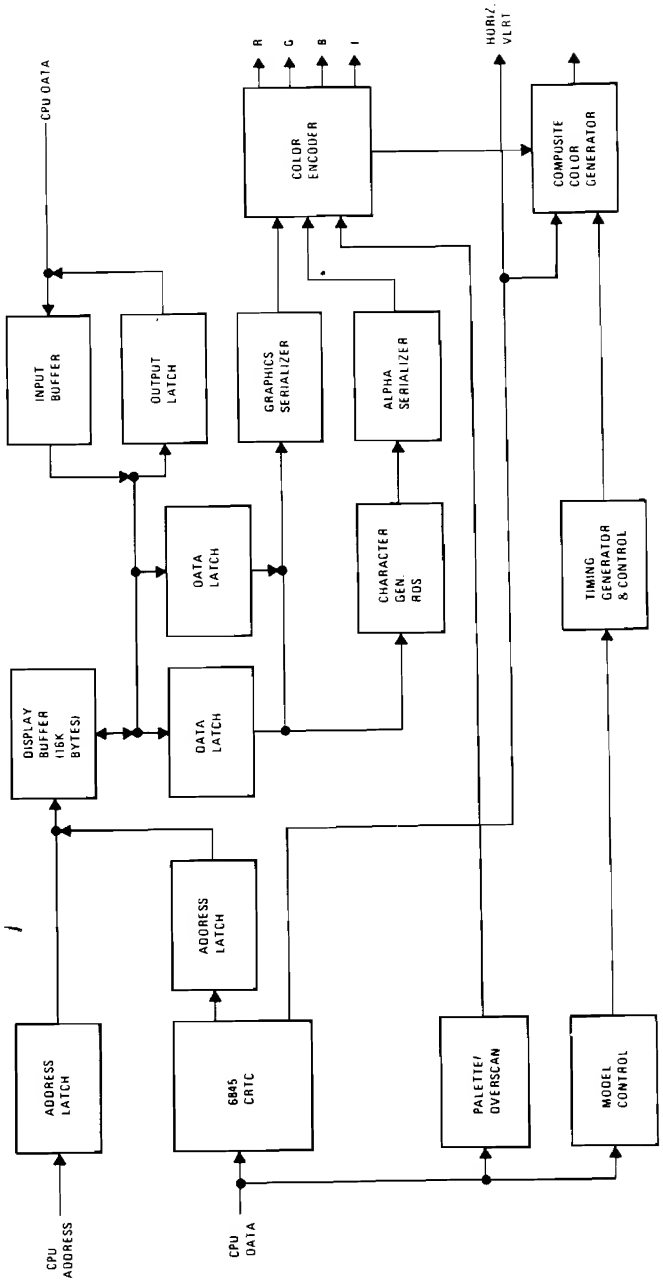


Figure 15. COLOR/GRAPHICS MONITOR ADAPTER BLOCK DIAGRAM

Major Components Definitions

Motorola 6845 CRT Controller

This device provides the necessary interface to drive a raster scan CRT.

Mode Set and Status Registers

This is a general purpose programmable I/O register. It has I/O points which may be individually programmed. Its function in this attachment is to provide mode selection (page 2-49 and 2-50) and color selection in the medium resolution color graphics mode (page 2-51).

Display Buffer

The Display Buffer resides in the CPU address space starting at address X'B8000'. It provides 16K bytes of dynamic read/write memory. A dual-ported implementation allows the CPU and the graphics control unit to access this buffer. The CPU and the CRT control unit have equal access to this buffer during all modes of operation except in high resolution alphanumeric mode. In this mode the CPU should access this buffer during the horizontal retrace intervals. The CPU may however, write to the required buffer at any time, but a small amount of display fetches will result if not during retrace intervals.

Character Generator

This attachment utilizes a ROM character generator. It consists of 8K bytes of storage which cannot be read/written under software control. This is a general purpose ROM character generator with three different character fonts. Two character fonts are used on this card (a 7x7 double dot and 5x7 single dot), selected by a card jumper. No jumper gives a 7x7 double dot; with a jumper a single dot font is selected.

Timing Generator

This block generates the timing signals used by the 6845 CRT controller and by the dynamic memory. It also resolves the CPU/graphic controller contentions for accessing the Display Buffer.

Composite Color Generator

The logic in this block generates base band video color information.

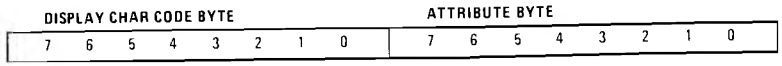
Modes of Operation

There are two basic modes of operation, 'Alphanumeric' and 'Graphics'. Each of these modes provide further options in both color and black-and-white. The following text describes each mode of operation.

Alphanumeric Mode

Alphanumeric Display Architecture

Every display character position is defined by two bytes in the regen buffer (part of display adapter, not system memory). Both the color and the black and white display adapter use this 2 byte character/attribute format.



Attribute Byte Definition

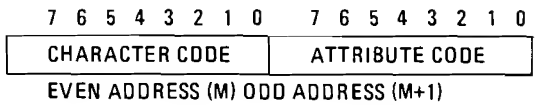
ATTRIBUTE BYTE

ATTRIBUTE FUNCTION	7	6	5	4	3	2	1	0
	B	R	G	B	I	R	G	B
	FG	BACKGROUND			FOREGROUND			
NORMAL	B	0	0	0	I	1	1	1
REVERSE VIDEO	B	1	1	1	I	0	0	0
NON DISPLAY (BLK)	B	0	0	0	I	0	0	0
NON DISPLAY (WHITE)	B	1	1	1	I	1	1	1

I = HIGH LIGHT FOREGROUND (CHAR)
B= BLINK FOREGROUND (CHAR)

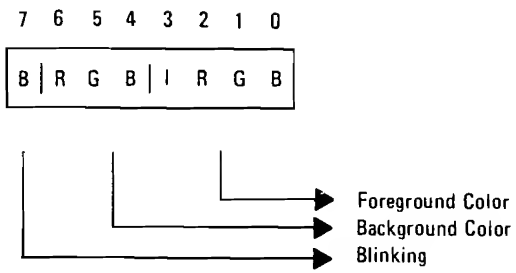
Color TV (NTSC - U.S. Standard)

- Display up to 25 rows of 40 characters each
- Maximum of 256 characters
- Requires 2000 bytes of Read/Write Memory (on the adapter)
- 8x8 character box
- 7x7 double dotted characters (one descender)
- Character attributes (one for each character)



ATTRIBUTE BYTE DEFINITIONS

R: Red
G: Green
B: Blue
I: Intensity



Note: The starting address of the display buffer must be an even location.

Color Monitor (with Drive input capability and 60 Hz vertical refresh)

Display up to 25 rows of 80 characters each

Requires 4000 bytes of Read/Write Memory (on the adapter)

Maximum of 256 character set

8x8 character box

7x7 character with one descender

Same format for attributes as for color TV

Note: The starting address of the display buffer must be an even location.

IBM Monochrome Display Adapter Vs. Color/Graphics Adapter Attribute Relationship

Table 3. Monochrome Vs. Color/Graphics Attributes

	7	6	5	4	3	2	1	0	ON THE MONOCHROME DISPLAY ADAPTER		ON THE COLOR/GRAPHIC DISPLAY ADAPTER	
	B	R	G	B	I	R	G	B	CHAR. COLOR	BKGD. COLOR	CHAR. COLOR	BKGD. COLOR
	FG	BACKGROUND			FOREGROUND							
NORMAL	B	0	0	0	1	1	1	1	WHITE	BLACK	WHITE	BLACK
RVV	B	1	1	1	1	0	0	0	BLACK	WHITE	BLACK	WHITE
NON DISP (BLK)	B	0	0	0	1	0	0	0	BLACK	BLACK	BLACK	BLACK
NON DISP (WHT)	B	1	1	1	1	1	1	1	WHITE	WHITE	WHITE	WHITE

ALL OTHER CODES DEFINE FOREGROUND BACKGROUND COLOR COMBINATIONS

ALL OTHER CODES RESULT IN WHITE CHAR ON BLACK BACKGROUND

ALL OTHER CODES CHANGE FOREGROUND BACKGROUND COLOR TO SELECTED VALUE

R	G	B		
				CODE WRITTEN WITH AN UNDERLINE ATTRIBUTE FOR THE IBM MONOCHROME DISPLAY
0	0	0	= BLACK	WHEN EXECUTED ON A COLOR/GRAPHICS ADAPTER
0	0	1	= BLUE	WILL RESULT IN A BLUE CHARACTER
0	1	0	= GREEN	WHERE THE UNDERLINE ATTRIBUTES
0	1	1	= CYAN	ARE ENCOUNTERED.
1	0	0	= RED	
1	0	1	= MAGENTA	CODE WRITTEN ON A COLOR/GRAPHICS ADAPTER
1	1	0	= YELLOW	WITH BLUE CHARACTERS, WILL BE
1	1	1	= WHITE	DISPLAYED AS WHITE CHARACTERS ON BLACK BACKGROUND WITH A WHITE UNDERLINE ON THE MONOCHROME DISPLAY

* AN ADDITIONAL 8 COLOR (ACTUAL) DIFFERENT SHADES OF THE ABOVE) ARE SELECTED BY SETTING THE (I) BIT

Note: Not all Monitors Recognize the (I) Bit

Table 4. Color/Graphics Modes

	HORIZONTAL	VERTICAL	NO OF COLORS (INCL. BACKGROUND COLOR)
LOW RES	160	100	16 (INCLUDES BLACK AND WHITE)
MED RES	320	200	4 COLORS: 1 OF 16 FOR BACKGROUND PLUS GREEN, RED, YELLOW OR CYAN, MAGENTA, WHITE
HIGH RES	640	200	B & W ONLY

1. Low resolution color graphics (TV or monitor).

   (Note: This mode is not supported in ROM).

   - Up to 100 rows of 160 pels each (2x2)
   - 1 of 16 colors each pel specified by I, R, G and B
   - Requires 8000 byte of Read/Write Memory (on the adapter)
   - Memory mapped graphics (requires special memory map and set up to be defined later)

2. Medium resolution color graphics (TV or monitor)

   - Up to 200 rows of 320 pels each (1x1)
   - 1 out of 4 preselected colors in each box
   - Requires 16000 bytes of Read/Write Memory (on the adapter)
   - Memory mapped graphics
   - 4 pels/byte

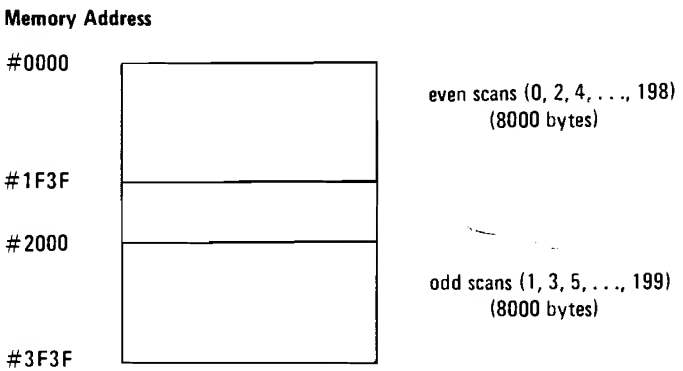
FORMAT:

7	6	5	4	3	2	1	0
C1	C0	C1	C0	C1	C0	C1	C0

First display pel

   - Graphics storage is organized in two banks of 8000 bytes each.

Graphics Storage Map



Address #0000 contains pel information for upper left corner of display area.

Color selection is determined by the following logic:

C1 and C0 will select 4 of 16 preselected colors.

This color selection (palette) is preloaded in an I/O port.

C1	C0	CODE SELECT COLOR FOR DISPLAY POSITION
0	0	DOT TAKES ON COLOR OF 1 OF 16 PRESELECTED BACKGROUND COLORS.
0	1	SELECT 1ST COLOR OF PRESELECT COLOR SET "1" OR "2"
1	0	SELECT 2ND COLOR OF PRESELECT COLOR SET "1" OR "2"
1	1	SELECT 3RD COLOR OF PRESELECT COLOR SET "1" OR "2"

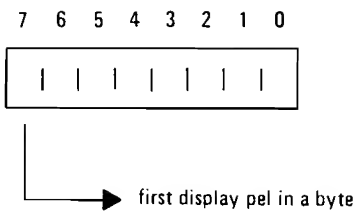
The two color sets are:

SET ONE	SET TWO
COLOR 1 – CYAN	COLOR 1 – GREEN
COLOR 2 – MAGENTA	COLOR 2 – RED
COLOR 3 – WHITE	COLOR 3 – BROWN

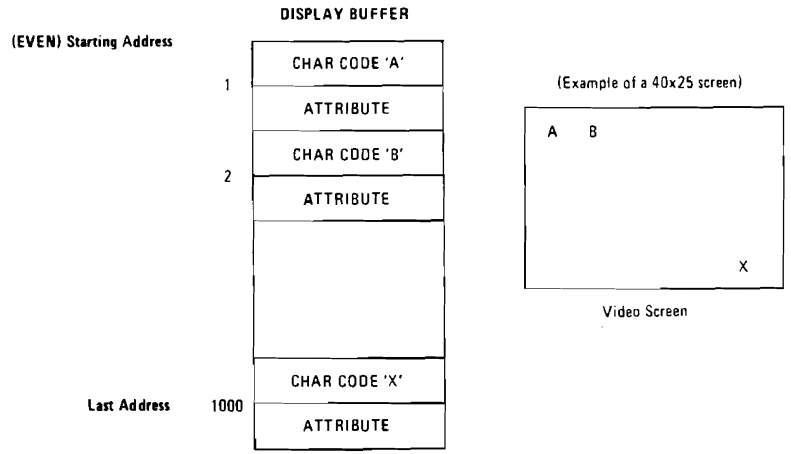
The background colors are the same basic 8 color as defined for low resolution graphic plus 8 alternate intensities defined by the intensity bit for a total of 16 color including black and white.

3. Black and white high resolution graphics (monitor)

   - Up to 200 rows of 640 pels each (1x1)
   - Black and white only
   - Requires 16000 bytes of Read/Write Memory (on the adapter)
   - Addressing and mapping is the same as for medium resolution color graphics, but the data format is different. In this mode each bit in memory is mapped to a pel on the screen.
   - 8 pels/byte



In the alphanumeric mode the adapter fetches character and attribute information from its display buffer. The starting address of the display buffer is programmable through the 6845, but it must be an even address. The character codes and attributes are then displayed according to their relative position in the buffer.



The CPU and the display control unit have equal access to the display buffer during all the operating modes except high resolution alphanumeric. During this mode, the CPU should access the display buffer during the vertical retrace time (if not, then the display will be affected with random patterns as the CPU is using the display buffer). The characters are displayed from a prestored "character generator" which contains the dot patterns of all the displayable characters.

In the graphics mode the displayed dots and colors are also fetched from the display buffer (up to 16K bytes). In the Color/Graphics Mode Section, the bit configuration for each graphics mode is explained.

Table 5. Summary of Available Colors

I	R	G	B	COLOR
0	0	0	0	Black
0	0	0	1	Blue
0	0	1	0	Green
0	0	1	1	Cyan
0	1	0	0	Red
0	1	0	1	Magenta
0	1	1	0	Brown
0	1	1	1	Light Gray
1	0	0	0	Dark Gray
1	0	0	1	Light Blue
1	0	1	0	Light Green
1	0	1	1	Light Cyan
1	1	0	0	Light Red
1	1	0	1	Light Magenta
1	1	1	0	Yellow
1	1	1	1	White

Note: "I" provides extra luminance (brightness) to each shade available. Resulting in the light colors listed above, except where the "I" bit is not recognized by some monitors.

Programming Considerations

Programming the 6845 CRT Controller

The 6845 has 19 internal registers which are used to define and control a raster scanned CRT display. One of these registers, the Address Register, is actually used as a pointer to the other 18 registers It is a write only register which is loaded from the CPU by executing an OUT instruction to I/O address 3D4. The five least significant bits of the I/O bus are loaded into the Address Register.

In order to load any of the other 18 registers, the Address Register is first loaded with the necessary pointer and then the CPU may output a value to I/O address 3D5 in order to load the information in the preselected register.

The following table defines the values which must be loaded in 6845 Registers in order to control the different modes of operation supported by the attachment.

Table 6. 6845 Register Description

ADDR REG.	REG. #	REGISTER TYPE	UNITS	I/O	40x25 ALPHA	80x25 ALPHA	GRAPHIC MODES
0	R0	Horizontal Total	Char.	Write Only	38	71	38
1	R1	Horizontal Displayed	Char.	Write Only	28	50	28
2	R2	Horiz. Sync Position	Char.	Write Only	2D	5A	2D
3	R3	Horiz. Sync Width	Char.	Write Only	0A	0A	0A
4	R4	Vertical Total	Char. Row	Write Only	1F	1F	7F
5	R5	Vertical Total Adjust	Scan Line	Write Only	06	06	06
6	R6	Vertical Displayed	Char. Row	Write Only	19	19	64
7	R7	Vert. Sync Position	Char. Row	Write Only	1C	1C	70
8	R8	Interlace Mode	–	Write Only	02	02	02
9	R9	Max Scan Line Addr.	Scan Line	Write Only	07	07	01
A	R10	Cursor Start	Scan Line	Write Only	06	06	06
B	R11	Cursor End	Scan Line	Write Only	07	07	07
C	R12	Start Addr. (H)	–	Write Only	00	00	00
D	R13	Start Addr. (L)	–	Write Only	00	00	00
E	R14	Cursor Addr. (H)	–	Read/ Write	XX	XX	XX
F	R15	Cursor Addr. (L)	–	Read/ Write	XX	XX	XX
10	R16	Light Pen (H)	–	Read Only	XX	XX	XX
11	R17	Light Pen (L)	–	Read Only	XX	XX	XX

Note: All register values are given in hexadecimal.

Programming the Mode Control and Status Register

The following I/O devices are defined on the Color/Graphics Adapter.

HEX ADDR.	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	FUNCTION OF REGISTER
X'3D8'	1	1	1	1	0	1	1	0	0	0	DD REG (MODE CONTROL)
X'3D9'	1	1	1	1	0	1	1	0	0	1	DD REG (COLOR SELECT)
X'3DA'	1	1	1	1	0	1	1	0	1	0	DI REG (STATUS)
X'3DB'	1	1	1	1	0	1	1	0	1	1	CLEAR LIGHT PEN LATCH
X'3DC'	1	1	1	1	0	1	1	1	0	0	PRE SET LIGHT PEN LATCH
X'3D0'	1	1	1	1	0	1	0	Z	Z	0	6845 REGISTERS
X'3D1'	1	1	1	1	0	1	0	Z	Z	1	6845 REGISTERS
X'3D0'	1	1	1	1	0	1	0	Z	Z	0	6845 REGISTERS
X'3D1'	1	1	1	1	0	1	0	Z	Z	1	6845 REGISTERS

Z = don't care condition

Color Select Register

This is a 6 bit output only, register, it cannot be read, its address is X'3D9' and can be written using the 8088 I/O OUT command.

The following is a description of the Register functions.

Bit 0	B (BLUE) Border Color Select ALPHA/BACKGROUND
Bit 1	G (GREEN) Border Color Select ALPHA/BACKGROUND
Bit 2	R (RED) Border Color Select ALPHA/BACKGROUND
Bit 3	I Intensifies Border Color Select ALPHA/BACKGROUND IN 320 x 200
Bit 4	Select Alt Back Color Set For Alpha Color Modes
Bit 5	320 x 200 Color Set Select
Bit 6	Not Used
Bit 7	Not Used

Bits 0, 1, 2, 3. Select the screens border color in 40x25 alpha mode. In graphics mode (medium resolution) 320x200 color, the screen background color (C0-C1) is selected by these bit settings.

Bit 4. This bit when set will select on alternate, intensified, set of background colors in the alpha mode.

Bit 5 is only used in the medium resolution color mode (320x200). It is used to select the active set of screen colors for the display.

When bit 5 is set to a "1" colors are determined as follows:

The C1	C0	Set selected are:
0	0	Background as defined by Bit 0-3 of Port '3D9'
0	1	Cyan
1	0	Magenta
1	1	White

When bit 5 is set to a "0" colors are determined as follows:

The C0	C1	Set selected are:
0	0	Background as defined by Bit 0-3 of Port '3D9'
0	1	Green
1	0	Red
1	0	Yellow

Mode Select Register

This is a 6 bit output only register, it cannot be read. Its address is X'3D8'. It can be written using the 8088 I/O OUT command.

The following is a description of the registers functions.

Bit 0	80x25 mode
Bit 1	Graphic Select
Bit 2	B & W Select
Bit 3	Enable Video Signal
Bit 4	High Res 640x200 B & W Mode
Bit 5	Change BACKGROUND INTENSITY to Blink Bit
Bit 6	Not Used
Bit 7	Not Used

Bit 0 Selects between 40x25 and 80x25 alpha mode, a "1" sets it to 80x25 mode.

Bit 1 Selects between ALPHA mode and 320x200 graphics mode, a "1" select 320x200 graphics mode.

Bit 2 Selects color or B & W mode, a "1" selects B & W.

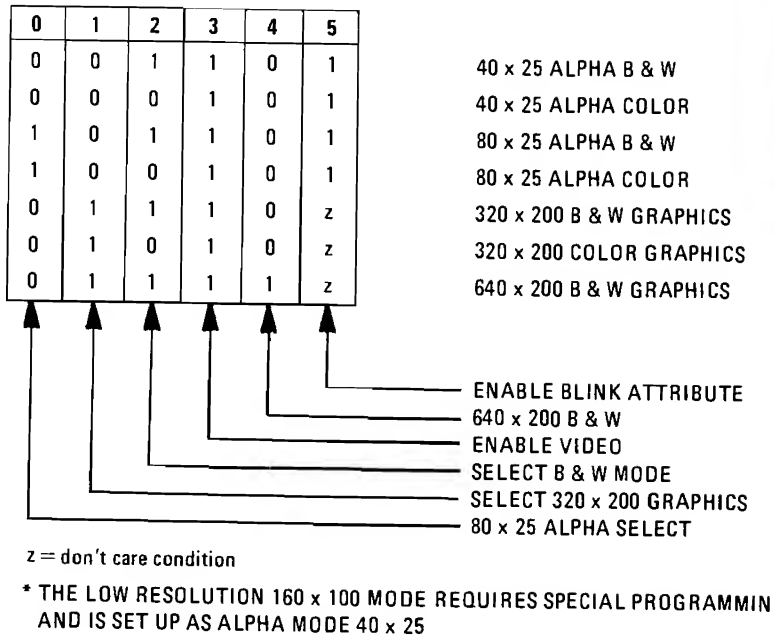
Bit 3 Enables the video signal at certain times when modes are being changed. The video signal should be disabled when changing modes. A "1" enables the video signal.

Bit 4 When on, this bit selects the 640x200 B & W graphics mode. One color of 8 can be selected on direct drive sets in this mode by using register 3D9.

Bit 5 When on, this bit will change the character background intensity to the blinking attribute function for ALPHA modes. When the high order attribute bit is not selected, 16 background colors (or intensified colors) are available. For normal operation, this bit should be set to "1" to allow the blinking function.

Mode Register Summary

Bits



z = don't care condition

* THE LOW RESOLUTION 160 x 100 MODE REQUIRES SPECIAL PROGRAMMING AND IS SET UP AS ALPHA MODE 40 x 25

Status Register

The status register is a 4 bit read only register. Its address is X'3DA'. It can be read using the 8088 I/O IN instruction.

The following is a description of the register functions.

Bit 0 Display Enable
Bit 1 Light Pen Trigger Set
Bit 2 Light Pen SW Made
Bit 3 Alpha Dots
Bit 4 Not Used
Bit 5 Not Used
Bit 6 Not Used
Bit 7 Not Used

Bit 0 This input bit, when active, indicates that a regen buffer memory access can be made without interfering with the Display.

Bit 1 This bit, when active, indicates that a positive going edge from the light pen input has set the light pen trigger. This trigger is reset on power on and may also be cleared by doing an I/O OUT command to address X'3DB'. No specific data setting is required, the action is address activated.

Bit 2 The light pen switch status is reflected in this status bit. The switch is not latched or debounced. A "0" indicates the switch is on.

Bit 3 The ALPHA video output signal is readable in this status bit. Its purpose is to verify that video information is being generated for RAS pusposes.

Sequence of Events

1. Determine mode of operation
2. Reset Video Enable bit
3. Program 6845 to select mode
4. Program mode/color select registers

Memory Requirements

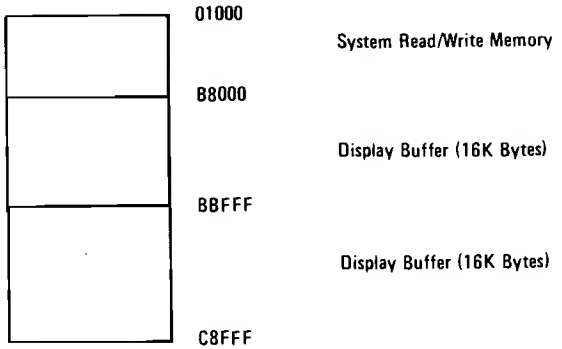
The memory used by this adapter is self-contained. It consists of 16K bytes of memory without parity. This memory is used as both a display buffer for alphanumeric data and as a bit map for graphics data. The Regen Buffers address starts at X'B8000'.

Interrupt Level (Vertical Retrace)

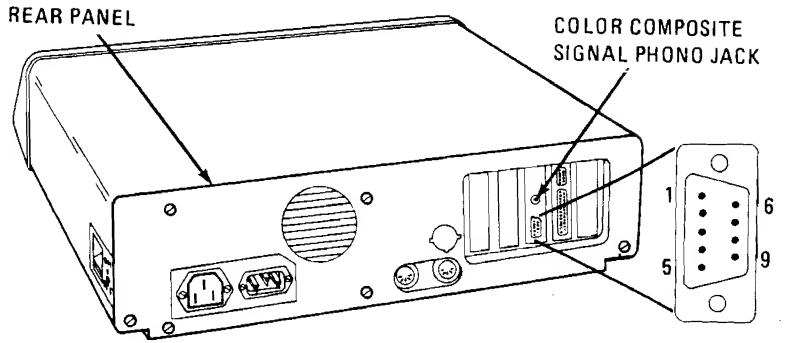
Level 2

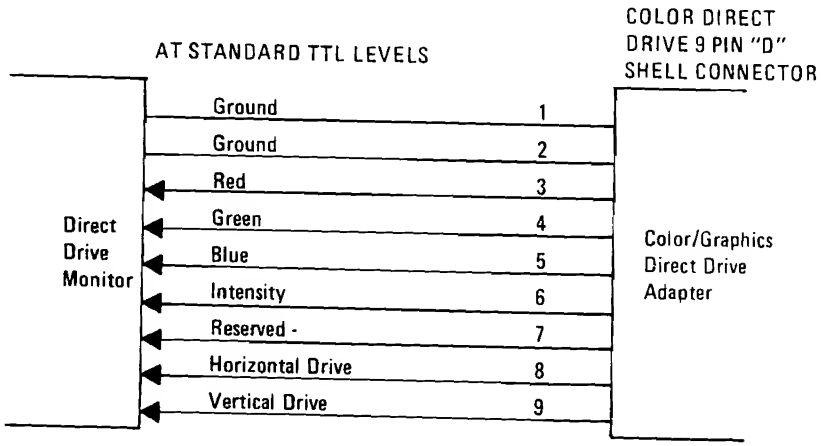
I/O Address and Bit Map

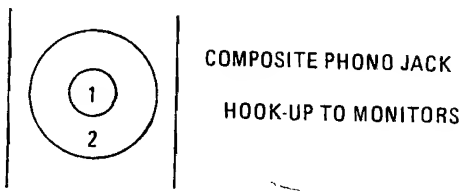
Read/Write Memory Address Space

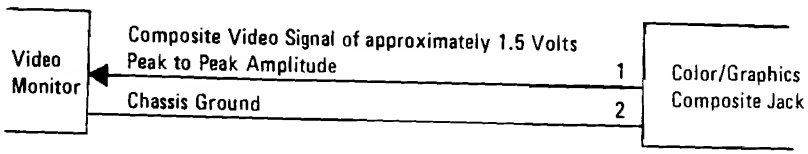


Color/Graphics Monitor Adapter Direct Drive and Composite Interface Pin Assignment

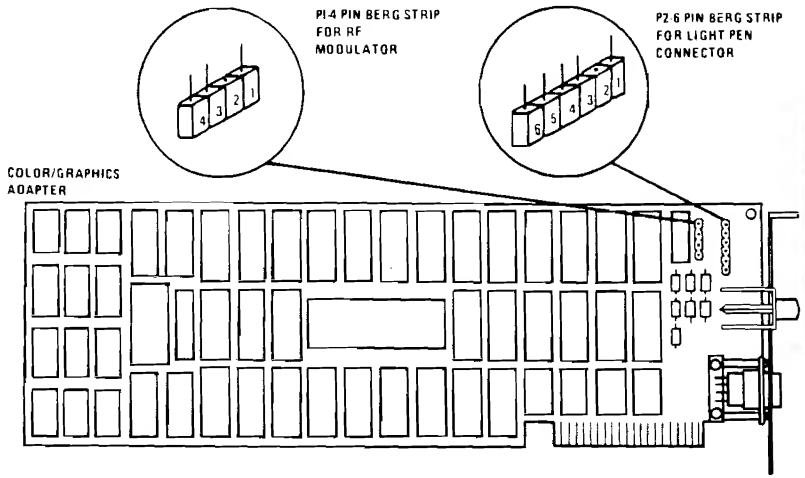


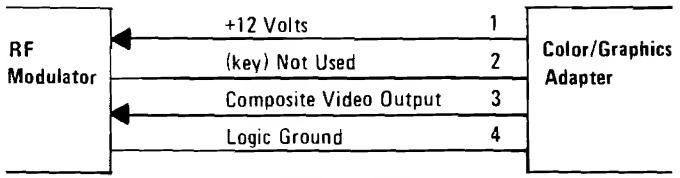




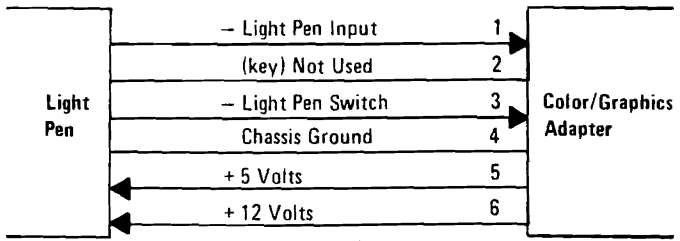


Color/Graphics Monitor Adapter Auxiliary Video Connectors





RF Modulator Interface



Light Pen Interface

NOTES

Parallel Printer Adapter

The Printer Adapter is specifically designed to attach printers with a parallel port interface, but it can be used as a general input/output port for any device or application which matches its input/output capabilities. It has 12 TTL buffer output points which are latched and can be written and read under program control using the processor IN or OUT instructions. The adapter also has five steady state input points that may be read using the processor's IN instructions.

In addition, one input can also be used to create a processor interrupt. This interrupt can be enabled and disabled under program control. Reset from the power-on circuit is also "ORed" with a program output point allowing a device to receive a power-on reset when the processor is reset.

This function is packaged on an adapter which fits into any of the five System Expansion slots on the System Board. The input/output signals are made available at the back of the adapter via a right angle PCB mounted 25 PIN "D" type connector. This connector protrudes through the rear panel of the System Unit where a cable and shield may be attached.

When this adapter is used to attach a printer, data, or printer, commands are loaded into an 8-bit latched output port, and the strobe line is activated writing data to the printer. The program then may read the input ports for printer status indicating when the next character can be written or it may use the interrupt line to indicate"not busy" to the software.

The output ports may also be read at the card's interface for diagnostic loop functions. This allows fault isolation determination between the adapter and the attaching device.

This same function is also part of the combination IBM Monochrome Display and Printer Adapter. A block diagram of the printer adapter is on the following page.

Parallel Printer Adapter Block Diagram

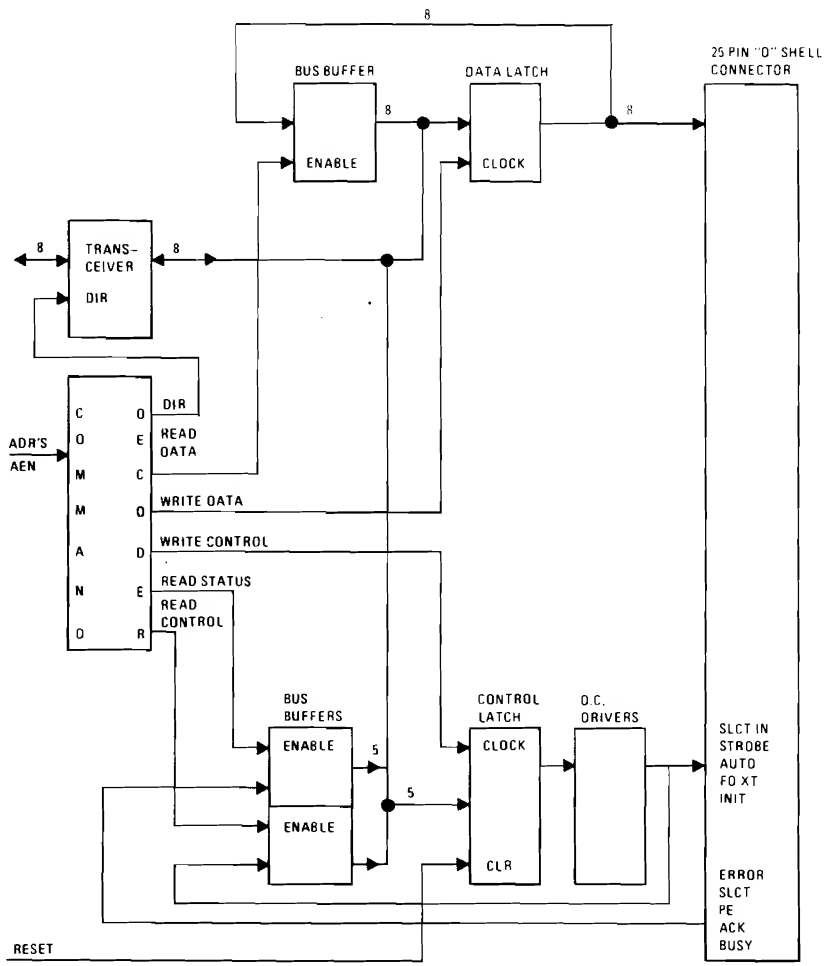


Figure 16. PARALLEL PRINTER ADAPTER BLOCK DIAGRAM

Programming Considerations

The Printer Adapter responds to 5 I/O instructions – 2 output and 3 input. The output instructions transfer data into 2 latches whose outputs are presented on pins of a 25 Pin "D" shell connector.

Two of the three input instructions allow the CPU to read back the contents of the two latches. The third allows the CPU to read the real time status of a group of pins on the connector.

A description of each instruction follows.

IBM Monochrome Display & Printer Adapter				Parallel Printer Adapter			
Output to address 3BCH				Output to address 378H			
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Pin 9	Pin 8	Pin 7	Pin 6	Pin 5	Pin 4	Pin 3	Pin 2

This instruction captures data from the data bus and is present on the respective pins. These pins are each capable of sourcing 2.6 ma and sinking 24 ma.

It is essential that the external device not try to pull these lines to ground.

IBM Monochrome Display & Printer Adapter		Parallel Printer Adapter			
Output to address 3BEH		Output to address 37AH			
	Bit 4	$\overline{\text{Bit 3}}$	Bit 2	$\overline{\text{Bit 1}}$	$\overline{\text{Bit 0}}$
	IRQ Enable	Pin 17	Pin 16	Pin 14	Pin 1

This instruction causes this latch to capture the five least significant bits of data bus. The four least significant bits present their outputs, or inverted versions of their outputs to the respective pins shown above. If bit 4 is written 1, the card will interrupt the CPU on the condition that Pin 10 transitions high to low.

These pins are driven by open collector drivers pulled to +5V through 4.7K OHM resistors. They can each sink approximately 7 ma and maintain 0.8 volts down level.

Note: For pin references, see Parallel Interface Connector Specifications, page 2-79.

IBM Monochrome Display & Printer Adapter	Parallel Printer Adapter
Input from address x' '3BC'	Input from address 378H

This command presents the CPU with data present on the pins associated with the out to x' '3BC'. This should normally reflect the exact value that was last written to x' '3BC'. If an external device should be driving data on these pins (in violation of usage ground rules) at the time of an input, this data will be 'or' ed with the latch contents.

IBM Monochrome Display & Printer Adapter	Parallel Printer Adapter
Input from address 3BDH	Input from address 379H

This command presents real time status to the CPU from the pins as follows.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Pin 11*	Pin 10	Pin 12	Pin 13	Pin 15	–	–	–

IBM Monochrome Display & Printer Adapter	Parallel Printer Adapter
Input from address 3BEH	Input from address 37AH

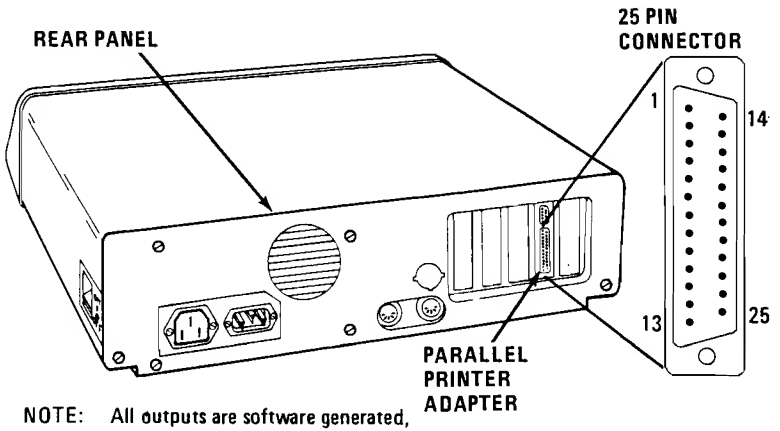
This instruction causes the data present on pins 1, 14, 16, 17 and IRQ bit to be read by the CPU. In the absence of external drive applied to these pins, data read by the CPU will exactly match data last written to x' '3BE' in the same bit positions. Note that data bits 0-2 are not included. If external drivers are dotted to these pins, that data will be 'or' ed with data applied to the pins by the x' '3BE' latch.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
			IRQ Enable	$\overline{\text{Pin 17}}$	Pin 16	$\overline{\text{Pin 14}}$	$\overline{\text{Pin 1}}$
			Por=0	Por=1	Por=0	Por=1	Por=1

These pins assume the states shown after a reset from the CPU.

Note: For pin references see Parallel Printer Adapter Interface Connector Specifications page 2-79.

Parallel Printer Adapter Interface Connector Specifications



NOTE: All outputs are software generated, and all inputs are real time signals (not latched).

AT STANDARD TTL LEVELS

Signal Name	Adapter Pin No.	
– Strobe	1	
+ Data Bit 0	2	
+ Data Bit 1	3	
+ Data Bit 2	4	
+ Data Bit 3	5	
+ Data Bit 4	6	
+ Data Bit 5	7	
+ Data Bit 6	8	
+ Data Bit 7	9	
– Acknowledge	10	
+ Busy	11	
+ P. End (out of Paper)	12	
+ Select	13	
– Auto Feed	14	
– Error	15	
– Initialize Printer	16	
– Select Input	17	
Ground	18 - 25	

Printer — Parallel Printer Adapter

IBM 80 CPS Graphics Printer

The printer is a self powered, standalone, table top unit. It attaches to the system unit via a parallel signal cable 6 feet (1.8 meters) in length. It obtains AC power from a standard wall outlet and is available in three different versions (120, 220, or 240 Vac). The printer is an 80 Character Per Second (CPS) bidirectional wire matrix device. It prints characters in a 9x9 dot matrix with a 9 wire head. It can print in a compressed mode of 132 characters per line and in a standard font, 80 characters per line. A large font also prints in 66 characters per line mode.

The printer can print double size characters and double strike characters. It can select from two character sets, an extended upper/lower case character set for international languages, and the U.S. standard upper/lower case 96 character ASCII character set. It can print subscript, superscript, defined graphics characters, underlining, and programmable graphics.

The printer can also accept commands setting the line feed control desired for the application. It attaches to the system unit via the Parallel Printer Adapter or the IBM Monochrome Display and Printer Adapter. The cable is a 25 lead shielded cable with a 25 pin "D" type connector at the system unit end, and a 36 pin connector on the printer end.

Table 7. Printer Specifications

(1)	Print Method:	Serial Impact dot matrix
(2)	Print Speed:	80 CPS
(3)	Print Direction:	Bidirectional with logical seeking
(4)	Number of Pins in Head	9
(5)	Line Spacing:	4.23 mm (1/6 inch) or programmable
(6)	Printing Characteristics	
	Matrix:	9x9
	Character Set 1:	Full 96-character ASCII with descenders, additional ASCII numbers 160 to 175 are European characters. 176 to 223 are graphic characters. 224 to 239 are selected Greek characters. 240 to 255 contain math and extra symbols.
	Character Set 2:	Same as above except: ASCII numbers 3,4,5, 6, and 21 contain symbols. ASCII numbers 128 to 175 contain European Characters.
	Graphics:	20 block characters and programmable graphics.
(7)	Printing Sizes	

	Characters per inch	Maximum characters per line
Normal:	10	80
Enlarged:	5	40
Condensed:	16.5	132
Condensed Enlarged:	8.25	66
Subscript or Superscript:	10	80

(8)	Media Handling	
	Paper Feed:	Adjustable sprocket pin feed
	Paper Width Range:	101.6 mm (4 inches) to 254 mm (10 inches)
	Copies:	One original plus two carbon copies, total thickness not to exceed 0.3 mm (0.012 inch). Minimum paper thickness is 0.064 mm or 0.0025 inch.
	Paper Path:	Rear
(9)	Interfaces	
	Standard:	Parallel 8-bit Data Data and Control lines
(10)	Inked Ribbon	
	Color:	Black
	Type:	Cartridge
	Life Expectancy:	3 million characters

NOTES

Table 7. Printer Specifications (continued)

(11)	Environmental Conditions			
	Operation Temperature Range:	5 to 35°C (41 to 95°F)		
	Operating Humidity:	10 to 80% non-condensing		
(12)	Power Requirements	Three models available:		
	Voltage (AC):	120	220	240
	Frequency (Hz):	60	50/60	50/60
	Maximum Current (Amps):	1	.5	.5
	Maximum Power (Watts):	100	100	100
(13)	Physical Characteristics			
	Height:	107 mm (4.2 inches)		
	Width:	374 mm (14.7 inches)		
	Depth:	305 mm (12.0 inches)		
	Weight:	5.5 kg (12 lbs.)		

Setting the DIP Switches

There are two DIP switches on the control circuit board. In order to suit the user's specific requirements, desired control modes are selectable by the DIP switches. The functions of the switches and their preset conditions at the time of shipment are as shown in Table 8 (DIP Switch 1) and Table 9 (DIP Switch 2).

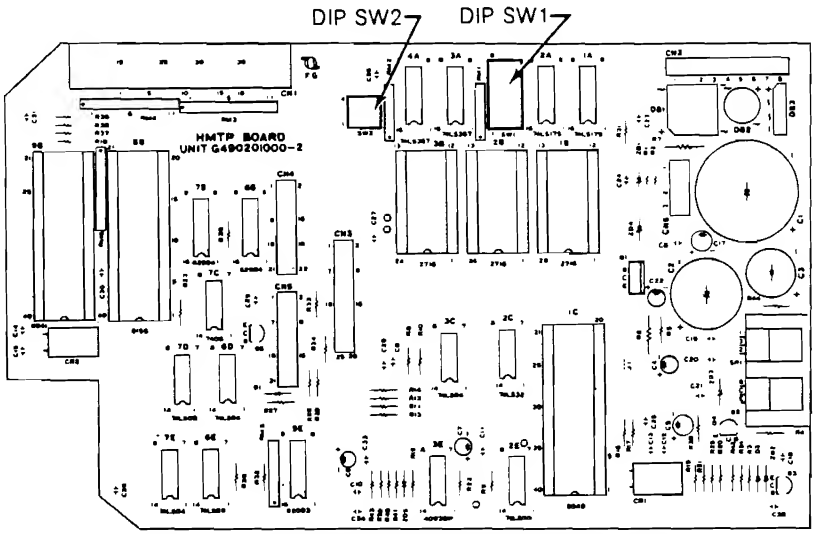


Figure 17. LOCATION OF PRINTER DIP SWITCHES

Table 8. Functions and Conditions of DIP Switch 1

Switch No.	Function	ON	OFF	Factory-set Condition
1-1	Not applicable	–	–	ON
1-2	CR	Print only	Print & line feed	ON
1-3	Buffer full	Print only	Print & line feed	OFF
1-4	Cancel code	Invalid	Valid	OFF
1-5	Not applicable	–	–	ON
1-6	Error buzzer	Sound	Does not sound	ON
1-7	Character generator	Table 2	Table 1	ON
1-8	Select in signal	Fixed internally	Not fixed internally	ON

Table 9. Functions and Conditions of DIP Switch 2

Switch No.	Function	ON	OFF	Factory-set Condition
2-1	Form length	12" 304.8 mm	11" 279.4 mm	OFF
2-2	Line spacing	1/8" 3.175 mm	1/6" 4.23 mm	OFF
2-3	Auto Feed XT signal	Fixed internally	Not fixed internally	OFF
2-4	1 inch skip over perforation	Valid	Not valid	OFF

Parallel Interface Description

(1) Specifications
  (a) Data transfer rate: 1000 CPS (max.)
  (b) Synchronization: By externally supplied STROBE pulses.
  (c) Handshaking: $\overline{\text{ACKNLG}}$ or BUSY signals.
  (d) Logic level: Input data and all interface control signals are compatible with the TTL level.

(2) Connector
Plug: 57-30360 (AMPHENOL)

(3) Connector pin assignment and descriptions of signals.
Connector pin assignment and descriptions of respective interface signals are provided in Table (10) page 2-86.

Table 10. Connector Pin Assignment and Description of Interface Signals

Signal Pin No.	Return Pin No.	Signal	Direction	Description
1	19	$\overline{\text{STROBE}}$	In	STROBE pulse to read data in. Pulse width must be more than 0.5μs at receiving terminal. The signal level is normally "HIGH"; read-in of data is performed at the "LOW" level of this signal.
2	20	DATA 1	In	These signals represent information of the 1st to 8th bits of parallel data respectively. Each signal is at "HIGH" level when data is logical "1" and "LOW" when logical "0"
3	21	DATA 2	In	
4	22	DATA 3	In	
5	23	DATA 4	In	
6	24	DATA 5	In	
7	25	DATA 6	In	
8	26	DATA 7	In	
9	27	DATA 8	In	
10	28	$\overline{\text{ACKNLG}}$	Out	Approx. 5μs pulse. "LOW" indicates that data has been received and that the printer is ready to accept other data.
11	29	BUSY	Out	A "HIGH" signal indicates that the printer cannot receive data. The signal becomes "High" in the following cases: 1. During data entry 2. During printing operation 3. In OFF-LINE state 4. During printer error status.

Table 10. Connector Pin Assignment and Description of Interface Signals (continued)

Signal Pin No.	Return Pin No.	Signal	Direction	Description
12	30	PE	Out	A "HIGH" signal indicates that the printer is out of paper.
13	–	SLCT	Out	This signal indicates that the printer is in the selected state.
14	–	AUTO FEED XT	In	With this signal being at "LOW" level, the paper is automatically fed one line after printing. (The signal level can be fixed to "LOW" with DIP SW pin 2-3 provided on the control circuit board.)
15	–	NC		Not used.
16	–	0V		Logic GND level.
17	–	CHASSIS-GND	–	Printer chassis GND. In the printer, the chassis GND and the logic GND are isolated from each other.
18	–	NC	–	Not used.
19–30	–	GND	–	TWISTED-PAIR RETURN signal GND level.
31	–	INIT	In	When the level of this signal becomes "LOW" the printer controller is reset to its initial state and the print buffer is cleared. This signal is normally at "HIGH" level, and its pulse width must be more than 50μs at the receiving terminal.

Table 10. Connector Pin Assignment and Description of Interface Signals (continued)

Signal Pin No.	Return Pin No.	Signal	Direction	Description
32		ERROR	Out	The level of this signal becomes "LOW" when the printer is in– 1. PAPER END state 2. OFF-LINE state 3. Error state
33	–	GND	–	Same as with Pin No. 19 to 30.
34	–	NC	–	Not used.
35				Pulled up to +5V through 4.7KΩ resistance.
36	–	$\overline{\text{SLCT IN}}$	In	Data entry to the printer is possible only when the level of this signal is "LOW". (Internal fixing can be carried out with DIP SW 1-8. The condition at the time of shipment is set "LOW" for this signal.)

NOTES 1: "Direction" refers to the direction of signal flow as viewed from the printer.
2: "Return" denotes "TWISTED PAIR RETURN" and is to be connected at signal ground level.
As to the wiring for the interface, be sure to use a twisted-pair cable for each signal and never fail to complete connection on the Return side. To prevent noise effectively, these cables should be shielded and connected to the chassis of the System Unit and the printer, respectively.
3: All interface conditions are based on TTL level. Both the rise and fall times of each signal must be less than 0.2μs.
4: Data transfer must not be carried out by ignoring the $\overline{\text{ACKNLG}}$ or BUSY signal. (Data transfer to this printer can be carried out only after confirming the $\overline{\text{ACKNLG}}$ signal or when the level of the BUSY signal is "LOW".)

(4) Data transfer sequence
Figure 17 shows the sequence for data transmission.

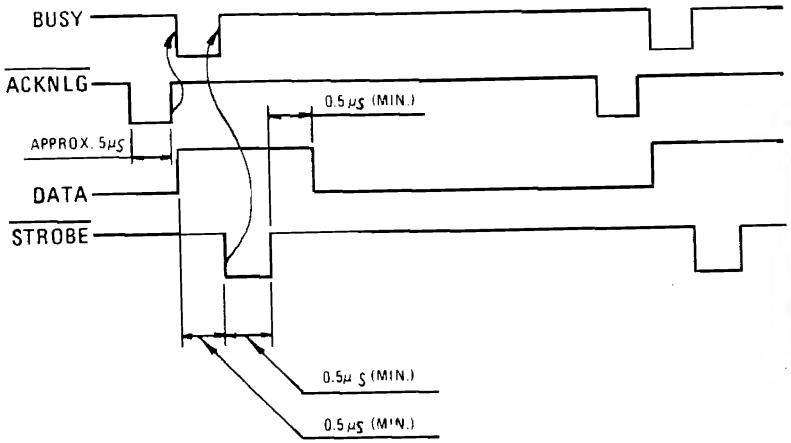


Figure 18. PARALLEL INTERFACE TIMING DIAGRAM

IBM 80 CPS Graphics Printer Modes

Below are the allowed combinations of print modes that can be selected.

The IBM 80 CPS Graphics Printer can select any of the combinations listed below and may change print modes at any place within a line.

Note: Modes can be selected and combined if they are in the same vertical column.

Normal	X	X	X								
Compressed					X	X	X				
Emphasized									X	X	X
Double Strike	X				X				X		
Subscript		X				X				X	
Superscript			X				X				X
Double Width	X	X	X		X	X	X		X	X	X
Underline	X	X	X		X	X	X		X	X	X

Printer Control Codes

On the following pages you will find complete codes for printer control, characters, and graphics. You may want to keep them handy for future reference. The printer codes are listed in ASCII decimal order (e.g. NUL is zero, BEL is 7, etc.). Examples given in the Printer Function descriptions are written in BASIC. The "Format" description is given when more information is needed for programming considerations. ASCII decimal values for the printer control codes can be found in the Printer Character Sets which follow these control codes.

Printer Code	Printer Function
NUL	Null Used with ESC B and ESC D as a list terminator. NUL is also used with other Printer Control Codes to select options (e.g. ESC S). Example: LPRINT CHR$(0);
BEL	Bell Sounds the printer buzzer for 1 second. Example: LPRINT CHR$(7);
HT	Horizontal Tab Tabs to the horizontal tab stop. When the printer is powered on or reset, tab stops are set every eight columns. Tab stops can be changed with the ESC D command. Example: LPRINT CHR$(9)
LF	Line Feed Prints data remaining in the printer buffer and spaces the paper up one line. Line spacing is 1/6 inch (4.23 mm) unless reset by ESC A, ESC 0, ESC 1, ESC 2, or ESC 3. Example: LPRINT CHR$(10);

Printer Code	Printer Function
VT	Vertical Tab The VT code is treated as a LF. Example: LPRINT CHR$(11);
FF	Form Feed Advances the paper to the top of the next page. The location of the paper, when the printer is powered on or reset, determines the top of the page. The next top of page is 11 inches (279.4 mm) from that position. ESC C can be used to change the page length. Example: LPRINT CHR$(12);
CR	Carriage Return Ends the line the printer is on and prints the data remaining in the printer buffer. (No line feed operation is initiated by the printer.) Example: LPRINT CHR$(13); IBM Personal Computer Basic adds a Line Feed command to every Carriage Return unless you select character set 1, add 128 to the CR, and return to character set 2. (See ESC 6 and ESC 7.) Example: LPRINT "0000";CHR$(27);CHR$(55);CHR$(13+128); CHR$(27);CHR$(54);"////"
SO	Shift Out (Double Width) Changes to the Double Width print mode. A Carriage Return, Line Feed, DC4, or CAN cancels Double Width print mode initiated with SO. See ESC W for continuous Double Width printing. Example: LPRINT CHR$(14);

Printer Code	Printer Function
SI	Shift In (Compressed) Changes the printer to the Compressed Character print mode. Example: LPRINT CHR$(15);
DC2	Device Control 2 (Compressed Off) Stops printing in the Compressed Character print mode. Example: LPRINT CHR$(18);
DC4	Device Control 4 (Double Width Off) Stops printing in the Double Width print mode started by the SO command. Example: LPRINT CHR$(20);
CAN	Cancel Clears the printer buffer without printing data. Control codes, except SO, remain in effect. Example: LPRINT CHR$(24);
ESC	Escape Lets the printer know that the next data sent is a printer command. (See the following list of commands.) Example: LPRINT CHR$(27);
ESC -	Escape Minus (Underline) Format: ESC -;n; ESC - followed by a 1, prints all of the following data with an underline. ESC - followed by a 0 (zero), cancels the Underline print mode. Example: LPRINT CHR$(27);CHR(45);CHR$(1);

Printer Code	Printer Function
ESC 0	Escape Zero (1/8-inch Line Feeding) Changes paper feeding to 1/8 inch (3.175 mm). Example: LPRINT CHR$(27);CHR$(48);
ESC 1	Escape 1 (7/72-inch Line Feeding) Changes paper feed to 7/72 inch (2.47 mm). Example: LPRINT CHR$(27);CHR$(49);
ESC 2	Escape Two (Starts Variable Line Feeding) ESC 2 is an execution command for ESC A. If no ESC A command has been given, line feeding returns to 1/6 inch (4.23 mm). Example: LPRINT CHR$(27);CHR$(50);
ESC 3	Escape Three (Variable Line Feeding) Format: ESC 3;n; (Graphics Printer only) Changes the paper feeding to n/216-inch (1/216 inch is 0.1176 mm). A value from 1 to 255 must be assigned to n. The example below sets the paper feeding to 54/216 (1/4) inch (6.35 mm). Example: LPRINT CHR$(27);CHR$(51);CHR$(54);
ESC 6	Escape Six (Select Character Set 2) Selects character set 2. (See Printer Character Set 2.) Character Set 2 is selected when the printer is powered on or reset. Example: LPRINT CHR$(27);CHR$(54);
ESC 7	Escape Seven (Select Character Set 1) Selects character set 1. (See Printer Character Set 1.) Example: LPRINT CHR$(27);CHR$(55);

Printer Code	Printer Function
ESC 8	Escape Eight (Ignore Paper End) Allows the printer to print to the end of the paper. The printer ignores the paper end switch. This command must be sent prior to an actual paper end. Example: LPRINT CHR$(27);CHR$(56);
ESC 9	Escape Nine (Cancel Ignore Paper End) Cancels the Ignore Paper End command. ESC 9 is selected when the printer is powered on or reset. Example: LPRINT CHR$(27);CHR$(57);
ESC <	Escape Less Than (Home Head) The print head will return to the left margin to print the line following ESC <. This will occur for one line only. Example: LPRINT CHR$(27);CHR$(60);
ESC A	Escape A (Sets the variable line feeding) Format: ESC A;n; Escape A sets the line feed feeding to n/72 inch (1/72 inch is 0.3528 mm). The example below tells the printer to set line feeding to 24/72 inch (8.47 mm). ESC 2 must be sent to the printer before the line feeding will change, e.g. ESC A;24 (text) ESC 2 (text). The text following ESC A;24 will space at the previously set line-feed increments. The text following ESC 2 will be printed with new line feed increments of 24/72 inch (8.47 mm). Any increment between 1/72 and 85/72 may be used. The default for this command at power on or reset is 6/72 inch (4.23 mm). Example: LPRINT CHR$(27);CHR$(65);CHR$(24);CHR$(27);CHR$(50);

Printer Code	Printer Function
ESC C	Escape C (Set lines per page) Format: ESC C;n; Sets the page length in number of lines. The ESC C command must have a value (from 1 to 127) following it to specify the number of lines desired on the page. The example below sets the page length to 55 lines. The printer defaults to 66 lines per page when powered on or reset. Example: LPRINT CHR$(27);CHR$(67);CHR$(55); Escape C (Set inches per page) Format: ESC C;n;m; Sets the length of the page in inches (one inch is 25.4 mm). This command requires a value of 0 (zero) for n and a value between 1 and 22 for m. Example: LPRINT CHR$(27);CHR$(67);CHR$(0);CHR$(12);
ESC D	Escape D (Set Horizontal Tab Stops) Format: ESC D;n_1;n_2;...n_k;NUL; Sets the horizontal tab stop positions. The example below shows the horizontal tab stop positions set at printer column positions of 10, 20 and 40. They are followed by CHR$(0), the NUL code. They must be in ascending numerical order as shown. Tab stops can be set between 1 and 80 in normal print mode. When in the compressed print mode, tab stops can be set up to 132. Double Width characters take up two column positions. The maximum number of tabs that can be set is 28. The HT [CHR$(9)] is used to execute a tab operation. Example: LPRINT CHR$(27);CHR$(68);CHR$(10);CHR$(20); CHR$(40);CHR$(0);

Printer Code	Printer Function
ESC E	Escape E (Emphasized) Changes the printer to the Emphasized print mode. The printer speed is reduced to half during the Emphasized print mode. Example: LPRINT CHR$(27);CHR$(69);
ESC F	Escape F (Emphasized Off) Stops printing in the Emphasized print mode. Example: LPRINT CHR$(27);CHR$(70);
ESC G	Escape G (Double Strike) Changes the printer to the Double Strike print mode. The paper is spaced 1/216 inch (0.1176 mm) before the second pass of the print head. Example: LPRINT CHR$(27);CHR$(71);
ESC H	Escape H (Double Strike Off) Stops printing in the Double Strike mode. Example: LPRINT CHR$(27);CHR$(72);
ESC J	Escape J (Feed Paper n/216 inch) Format: ESC J;n; When ESC J is sent to the printer, the paper will be fed n/216 of an inch. (1/216 inch is 0.1176 mm.) The value of "n" must be between 1 and 255. The example below produces a paper feed of 50/216 inch (5.88 mm). ESC J is canceled after the paper feed takes place. Example: LPRINT CHR$(27);CHR$(74);CHR$(50);

Printer Code	Printer Function
ESC K	Escape K (480 Bit Image Graphics Mode) Format: ESC K;n_1 ;n_2 ;v_1 ;v_2 ;...v_k ; Changes from the Text mode to the Bit Image Graphics mode. n_1 and n_2 are numbers, each consisting of 1 byte, which specify the total number of Bit Image Data bytes to be transferred. v_1 through v_k are the bytes of Bit Image Data whose total number (k) cannot exceed 480 and must be equal to $n_1 + 256n_2$. At every horizontal position each byte can print up to 8 vertical dots. Bit Image Data may be mixed with Text data on the same line. **Note:** Assign values to n_1 and n_2 as follows: n_1 represents values from 0 - 255. n_2 represents values from 0 - 1 x 256. MSB is Most Significant Bit and LSB is Least Significant Bit.

n_2

MSB							LSB
2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8

n_1

MSB							LSB
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Data sent to the printer.

Text (20 characters)	ESC K n=360	Bit-image data	Next data

20 characters in text mode correspond to 120 bit-image positions (20 x 6 = 120). The printable portion left in Bit-image mode is 360 dot positions (480 - 120 = 360).

Data sent to the printer.

			n_1 , n_2				n_1 , n_2
Data A	ESC K	n_1 n_2	Data B	Data C	ESC K	n_1 n_2	Data D
Text data		Length of data	Bit-image data	Text data		Length of data	Bit-image data
← 480 bit-image dot positions →							

Example:

```
1   'OPEN PRINTER IN RANDOM MODE WITH LENGTH OF
    255
2   OPEN "LPT1:" AS #1
3   WIDTH "LPT1:",255
4   PRINT #1,CHR$(13);CHR$(10);
5   SLASH$=CHR$(1)+CHR$(2)+CHR$(4)+CHR$(8)
6   SLASH$=SLASH$+CHR$(16)+CHR$(32)+CHR$(64)
    +CHR$(128)+CHR$(0)
7   GAP$=CHR$(0)+CHR$(0)+CHR$(0)
8   NDOTS=480
9   'ESC K N1 N2
10  PRINT #1,CHR$(27);"K";CHR$(NDOTS MOD 256);
    CHR$(INT(NDOTS/256));
11  'SEND NDOTS NUMBER OF BIT IMAGE BYTES
12  FOR I=1 TO NDOTS/12 'NUMBER OF SLASHES TO
    PRINT USING GRAPHICS
13  PRINT #1,SLASH$;GAP$;
14  NEXT I
15  CLOSE
16  END
```

This example will give you a row of slashes printed in the 480 Bit Image mode.

Printer Code	Printer Function
ESC L	Escape L (960 Bit Image Graphics Mode) Format: ESC L;n_1 ;n_2 ;v_1 ;v_2 ;...v_k ; Changes from the Text mode to the Bit Image Graphics mode. The input is similar to ESC K. The 960 Bit Image mode prints at half the speed of the 480 Bit Image mode, but can produce a denser graphic image. The number of bytes of Bit Image Data (k) is $n_1 + 256n_2$ but cannot exceed 960. n_1 is in the range of 0 to 255.
ESC N	Escape N (Set Skip Perforation) Format: ESC N;n; Sets the Skip Perforation function between pages. The number following ESC N sets the value for the number of lines of Skip Perforation. This value can be from 1 to 127. The example shows a 12-line skip perforation. With a page length of 66 lines this will print 54 lines and feed the paper 12 lines. ESC N is reset anytime the page length (ESC C) is changed. Example: LPRINT CHR$(27);CHR$(78);CHR$(12);
ESC O	Escape O (Cancel Skip Perforation) Cancels the Skip Perforation function. Example: LPRINT CHR$(27);CHR$(79);
ESC S	Escape S (Subscript/Superscript) Format: ESC S;n; Changes the printer to the Subscript print mode when ESC S is followed by a 1 as in the example below. When ESC S is followed by a 0 (zero), the printer will print in the Superscript print mode. Example: LPRINT CHR$(27);CHR$(83);CHR$(1);

Printer Code	Printer Function
ESC T	Escape T (Subscript/Superscript Off) The printer stops printing in the Subscript or Superscript print mode. Example: LPRINT CHR$(27);CHR$(84);
ESC U	Escape U (Unidirectional Printing) Format: ESC U;n; The printer will print only from left to right following the input of ESC U;1. When ESC U is followed by a 0 (zero), the left to right printing operation is canceled. The Unidirectional print mode (ESC U) assures a more accurate printing start position for better print quality. Example: LPRINT CHR$(27);CHR$(85);CHR$(1);
ESC W	Escape W (Double Width) Format: ESC W;n; Changes to the Double Width mode when ESC W is followed by a 1. This mode is not canceled by a line feed operation or by the DC4 code. It must be canceled with ESC W followed by a 0 (zero). Example: LPRINT CHR$(27);CHR$(87);CHR$(1);
ESC Y	Escape Y (960 Bit Image Graphics Mode) Format: ESC Y n_1 ;n_2 ;v_1 ;v_2 ;...v_k ; Changes from the Text mode to the 960 Bit Image Graphics mode. The printer prints at normal speed during this operation and cannot print dots on consecutive dot positions. The input of data is similar to ESC L.
ESC Z	Escape Z (1920 Bit Image Graphics Mode) Format: ESC Z;n_1 ;n_2 ;v_1 ;v_2 ;...v_k ; Changes from the Text mode to the 1920 Bit Image Graphics mode. The input is similar to the other Bit Image Graphics modes during this operation. ESC Z prints at normal speed but can only print every third dot position.

Table 11. Graphics Printer Character Set 1

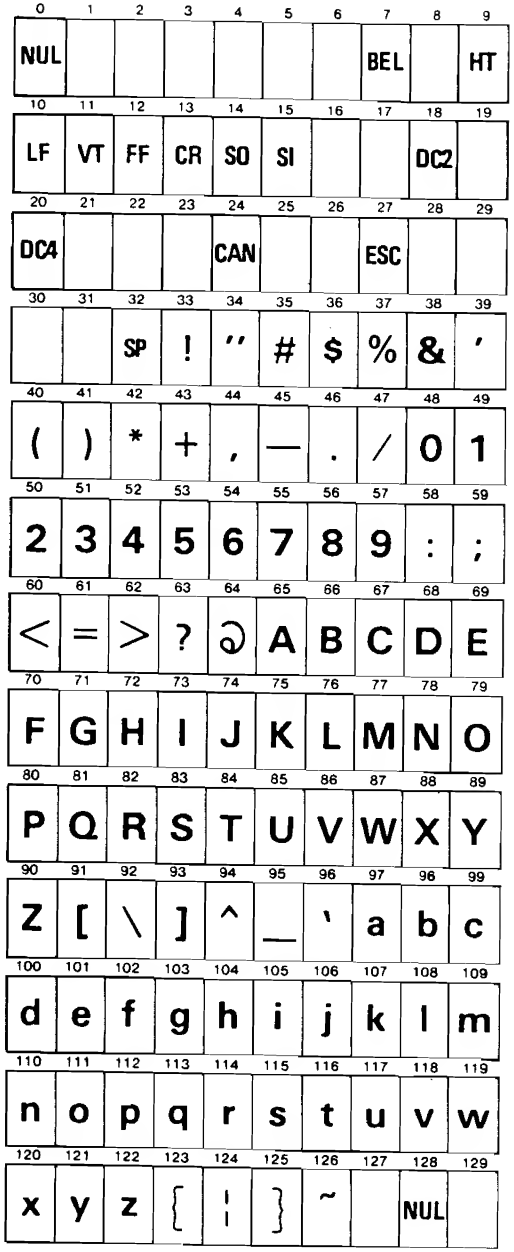


Table 11. Graphics Printer Character Set 1 (continued)

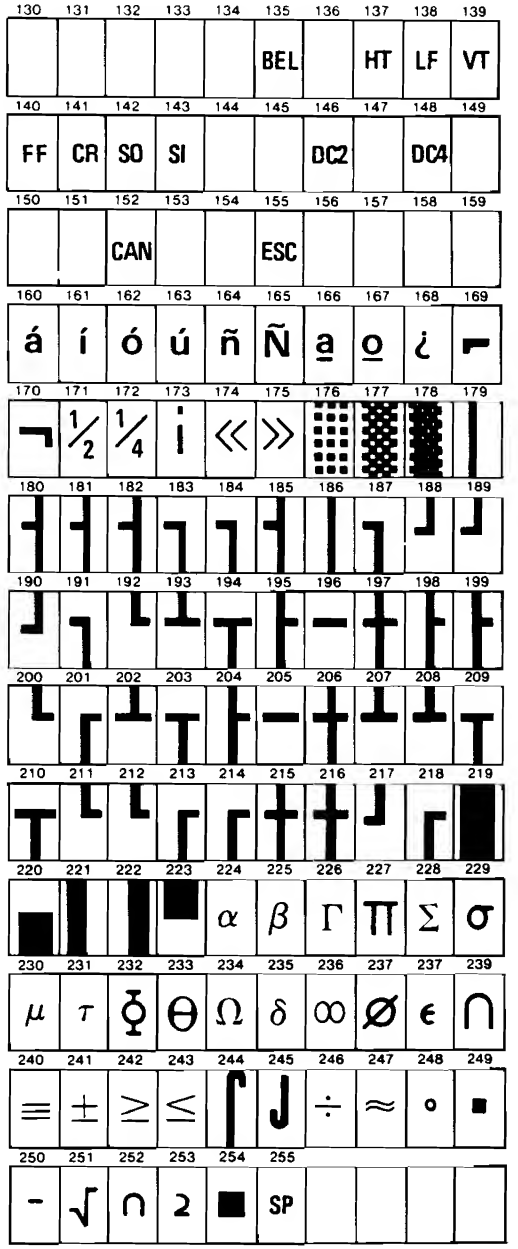


Table 12. Graphics Printer Character Set 2

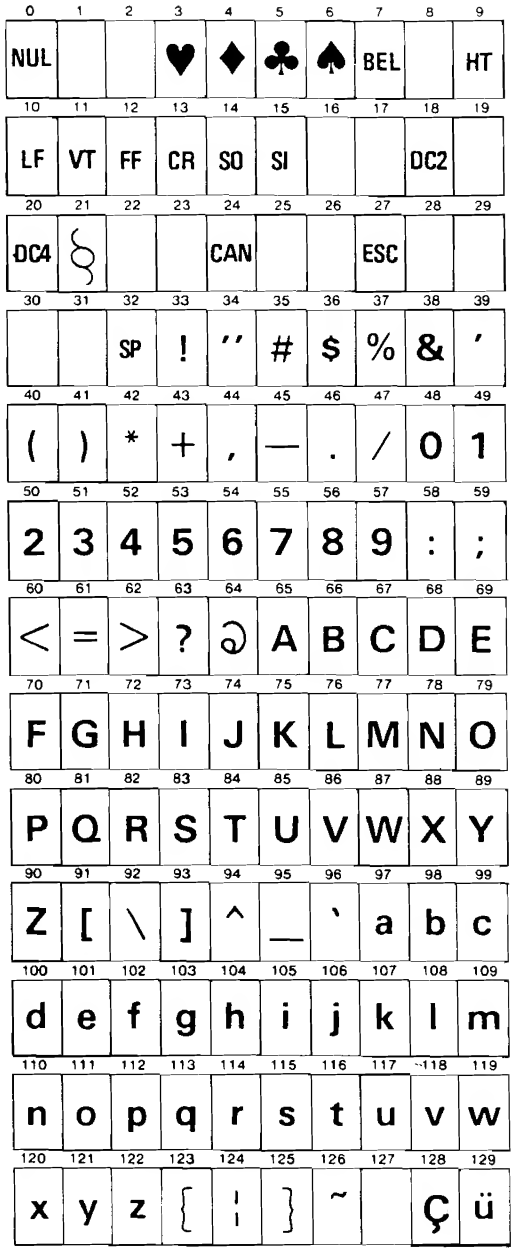
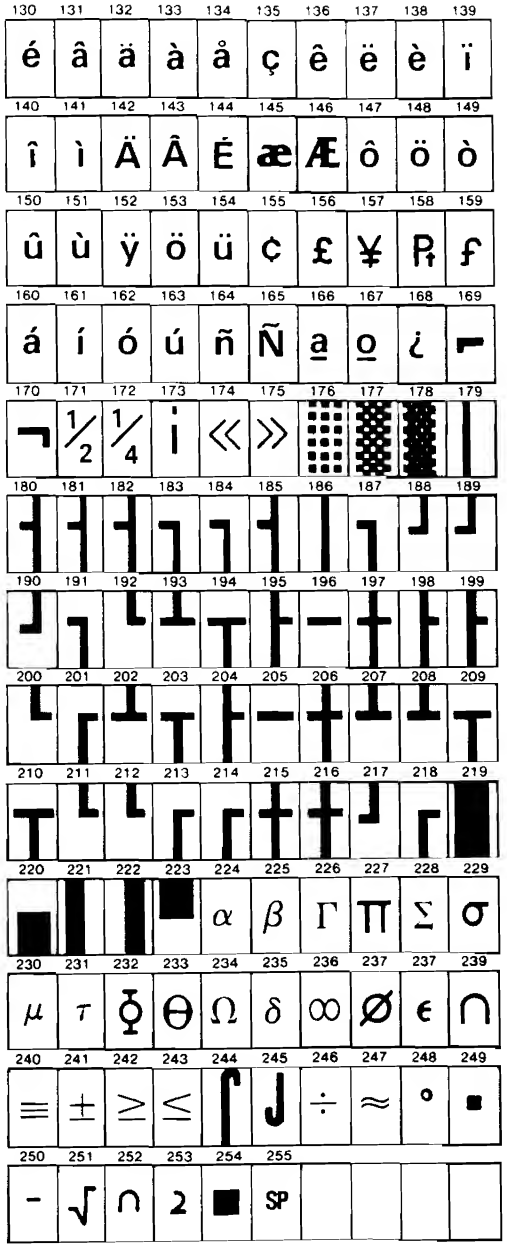


Table 12. Graphics Printer Character Set 2 (continued)



NOTES

5¼-Inch Diskette Drive Adapter

The System Unit has space and power for one or two 5-1/4" Diskette Drives. The drives are soft sectored, single or double sided, with 40 tracks per side. They are Modified Frequency Modulation (MFM) coded in 512 byte sectors, giving a formatted capacity of 163,840 bytes per drive for single sided and 327,680 bytes per drive for double sided. They have a track to track access time of 8 ms and a motor start time of 500 ms.

The 5-1/4" Diskette Drive Adapter fits in one of the System Board's five System Expansion Slots. It attaches to the two drives via an internal daisy chained flat cable which connects to one end of the drive adapter. The adapter has a second connector on the other end which extends through the rear panel of the System Unit. This connector contains the signals for two additional external drives, thus the 5-1/4" Diskette Drive Adapter is capable of attaching four 5-1/4" drives, two internal, and two external.

The adapter is designed for double density MFM coded drives and uses write precompensation with an analog phase locked loop for clock and data recovery. The adapter is a general purpose device using the NEC μPD765 compatible controller. Thus the drive parameters are programmable. In addition, the attachment supports the drive's write protect feature.

The adapter is buffered on the I/O bus and uses the System Board direct memory access (DMA) for record data transfers. An interrupt level is also used to indicate operation complete and status condition requiring processor attention.

In general, the 5-1/4" Diskette Drive Adapter presents a high-level command interface to software I/O drivers. A block diagram of the 5-1/4" Diskette Drive Adapter is on the following page.

5-1/4" Diskette Drive Adapter Block Diagram

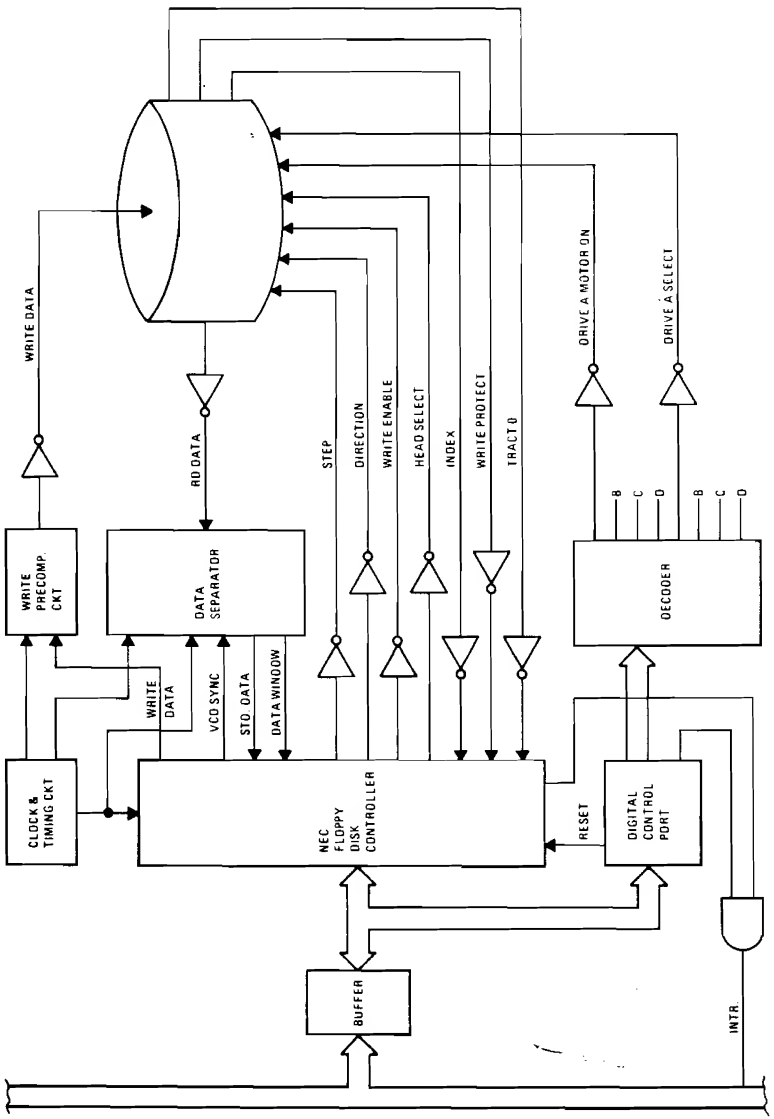


Figure 19. 5¼" DISKETTE DRIVE ADAPTER BLOCK DIAGRAM

Functional Description

From a programming point of view, this attachment consists of an 8-bit digital output register in parallel with a NEC μPD765 or equivalent Floppy Disk Controller (FDC).

In the following description, drives numbers 0-3 are equivalent to drives A-D respectively.

Digital Output Register (DOR)

The Digital Output Register (DOR) is an output only register used to control drive motors, drive selection, and feature enable. All bits are cleared by the I/O interface reset line. The bits have the following functions:

Bits 0 and 1 — These bits are decoded by the hardware to select one drive if its motor is on:

Bit 1	Bit 0	Drive	
0	0	0	A
0	1	1	B
1	0	2	C
1	1	3	D

Bit 2 — The FDC is held reset when this bit is clear. It must be set by the program to enable the FDC.

Bit 3 — This bit allows the FDC interrupt and DMA requests to be gated onto the I/O interface. If this bit is cleared, the interrupt and DMA request I/O interface drivers are disabled.

Bits 4,5,6, and 7 — These bits control respectively the motors of drives 0,1,2,A,B,C, and 3,D. If a bit is clear, the associated motor is off, and the drive cannot be selected.

Floppy Disk Controller (FDC)

The following is a brief summary of the registers and commands implemented by the FDC.

The FDC contains two registers which may be accessed by the main system processor; a Status Register and a Data Register. The 8-bit Main Status Register contains the status information of the FDC, and may be accessed at any time. The 8-bit Data Register (actually consisting of several registers in a stack with only one register presented to the data bus at a time) stores data, commands, parameters, and FDD status information. Data bytes are read out of, or written into, the Data Register in order to

program or obtain the results after a particular command. The Main Status Register may only be read and is used to facilitate the transfer of data between the processor and FDC.

The bits in the Main Status Register are defined as follows:

Bit Number	Name	Symbol	Description
DB0 FDD	FDD A Busy	DAB	FDD number is in the Seek mode.
DB1	FDD B Busy	DBB	FDD number 1 is in the Seek mode.
DB2	FDD C Busy	DCB	FDD number 2 is in the Seek mode.
DB3	FDD D Busy	DDB	FDD number 3 is in the Seek mode.
DB4	FDC Busy	CB	A read or write command is in process.
DB5	Non-DMA Mode	NDM	The FDC is in the non-DMA mode.
DB6	Data Input/	DIO	Indicates direction of data transfer between FDC and Processor. If DIO = "1", then transfer is from FDC Data Register to the Processor. If DIO = "0", then transfer is from the Processor to FDC Data Register.
DB7	Request for Master	RQM	Indicates Data Register is ready to send or receive data to or from the Processor. Both bits DIO and RQM should be used to perform the handshaking functions of "ready" and "direction" to the processor.

The FDC is capable of performing 15 different commands. Each command is initiated by a multi-byte transfer from the processor, and the result after execution of the command may also be a multi-byte transfer back to the processor. Because of this multi-byte interchange of information between the FDC and the processor, it is convenient to consider each command as consisting of three phases:

Command Phase

The FDC receives all information required to perform a particular operation from the processor.

Execution Phase

The FDC performs the operation it was instructed to do.

Result Phase

After completion of the operation, status and other housekeeping information are made available to the processor.

Programming Considerations

Table 13. Symbol Description

The following tables define the symbols used in the command summary which follows.

SYMBOL	NAME	DESCRIPTION
A0	Address Line 0	A0 controls selection of Main Status Register (A0 = 0) or Data Register (A0 = 1).
C	Cylinder Number	C stands for the current/selected Cylinder (track) number of the medium.
D	Data	D stands for the data pattern which is going to be written into a Sector.
D7-D0	Data Bus	8-bit Data Bus, where D7 stands for a most significant bit, and D0 stands for a least significant bit.
DTL	Data Length	When N is defined as 00, DTL stands for the data length which users are going to read out or write into the Sector.
EOT	End of Track	EOT stands for the final Sector number on a Cylinder.
GPL	Gap Length	GPL stands for the length of Gap 3 (spacing between Sectors excluding VCO Sync. Field).
H	Head Address	H stands for head number 0 or 1, as specified in ID field.
HD	Head	HD stands for a selected head number 0 or 1. (H = HD in all command words.)
HLT	Head Load Time	HLT stands for the head load time in the FDD (4 to 512 ms in 4 ms increments).
HUT	Head Unload Time	HUT stands for the head unload time after a read or write operation has occurred (0 to 480 ms in 32 ms increments.)
MF	FM or MFM Mode	If MF is low, FM mode is selected, and if it is high, MFM mode is selected only if MFM is implemented.
MT	Multi-Track	If MT is high, a multi-track operation is to be performed. (A cylinder under both HD0 and HD1 will be read or written.)
N	Number	N stands for the number of data bytes written in a Sector.
NCN	New Cylinder Number	NCN stands for a new Cylinder number, which is going to be reached as a result of the Seek operation. Desired position of Head.

Table 13. Symbol Descriptions (continued)

SYMBOL	NAME	DESCRIPTION
ND	Non-DMA Mode	ND stands for operation in the Non-DMA Mode.
PCN	Present Cylinder Number	PCN stands for Cylinder number at the completion of SENSE INTERRUPT STATUS Command, indicating the position of the Head at present time.
R	Record	R stands for the Sector number, which will be read or written.
R/W	Read/Write	R/W stands for either Read (R) or Write (W) signal.
SC	Sector	SC indicates the number of Sectors per Cylinder.
SK	Skip	SK stands for Skip Deleted Data Address Mark.
SRT	Step Rate Time	SRT stands for the Stepping Rate for the FDD. (2 to 32 ms in 2 ms increments.)
ST 0 ST 1 ST 2 ST 3	Status 0 Status 1 Status 2 Status 3	ST 0–3 stand for one of four registers which store the status information after a command has been executed. This information is available during the result phase after command execution. These registers should not be confused with the main status register (selected by A0 = 0). ST 0–3 may be read only after a command has been executed and contain information relevant to that particular command.
STP	Scan Test	During a Scan operation, if STP = 1, the data in contiguous sectors is compared byte by byte with data sent from the processor (or DMA), and if STP = 2, then alternate sectors are read and compared.
US0, US1	Unit Select	US stands for a selected drive number encoded the same as bits 0 and 1 of the digital register (DOR) p 2-91

Command Summary

0 indicates 'logical 0' for that bit, 1 means 'logical 1', X means 'don't care'.

PHASE	R/W	DATA BUS D7	D6	D5	D4	D3	D2	D1	D0	REMARKS
		READ DATA								
Command	W	MT	MF	SK	0	0	1	1	0	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W				C					Sector ID
	W				H					information prior
	W				R					to Command execution
	W				N					
	W				EOT					
	W				GPL					
	W				DTL					
Execution										Data-transfer between the FDD and main-system
Result	R				ST 0					Status information after
	R				ST 1					Command execution
	R				ST 2					
	R				C					Sector ID information
	R				H					after Command execution
	R				R					
	R				N					
		READ DELETED DATA								
Command	W	MT	MF	SK	0	1	1	0	0	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W				C					Sector ID information
	W				H					prior to Command
	W				R					execution
	W				N					
	W				EOT					
	W				GPL					
	W				DTL					
Execution										Data-transfer between the FDD and main-system
Result	R				ST 0					Status information after
	R				ST 1					command execution
	R				ST 2					
	R				C					Sector ID information
	R				H					after command execution
	R				R					
	R				N					
		WRITE DATA								
Command	W	MT	MF	0	0	0	1	0	1	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W				C					Sector ID information
	W				H					to command execution
	W				R					
	W				N					
	W				EOT					
	W				GPL					
	W				DTL					
Execution										Data-transfer between the main-system and FDD
Result	R				ST 0					Status information after
	R				ST 1					command execution
	R				ST 2					
	R				C					Sector ID information
	R				H					after command execution
	R				R					
	R				N					

Command Summary (continued)

PHASE	R/W	DATA BUS D7	D6	D5	D4	D3	D2	D1	D0	REMARKS
		WRITE DELETED DATA								
Command	W	MT	MF	0	0	1	0	0	1	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W				C					Sector ID information
	W				H					prior to command
	W				R					execution
	W				N					
	W				EOT					
	W				GPL					
	W				DTL					
Execution										Data-transfer between FDD and main-system
Result	R				ST 0					Status ID information
	R				ST 1					after common execution
	R				ST 2					
	R				C					Sector ID information
	R				H					after command execution
	R				R					
	R				N					
		READ A TRACK								
Command	W	0	MF	SK	0	0	0	1	0	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W				C					Sector ID information
	W				H					prior to command
	W				R					execution
	W				N					
	W				EOT					
	W				GPL					
	W				DTL					
Execution										Data-transfer between the FDD and main-system. FDC reads all of cylinders contents from index hole to EOT.
Result	R				ST 0					Status information after
	R				ST 1					command execution
	R				ST 2					
	R				C					Sector ID information
	R				H					after command execution
	R				R					
	R				N					
		READ ID								
Command	W	0	MF	0	0	1	0	1	0	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
Execution										The first correct ID information on the cylinder is stored in data register.
Result	R				ST 0					Status information
	R				ST 1					after command execution
	R				ST 2					
	R				C					Sector ID information
	R				H					during execution phase
	R				R					
	R				N					

PHASE	R/W	D7	D6	D5	D4	D3	D2	D1	D0	REMARKS
					FORMAT A TRACK					
Command	W	0	MF	0	0	1	1	0	0	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W				N					Bytes/Sector
	W				SC					Sector/Track
	W				GPL					Gap 3
	W				D					filler byte
Execution										FDC formats an entire cylinder
Result	R				ST 0					Status information
	R				ST 1					after command
	R				ST 2					execution
	R				C					In this case, the ID
	R				H					information has no
	R				R					meaning
	R				N					
					SCAN EQUAL					
Command	W	MT	MF	SK	1	0	0	0	1	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W				C					Sector ID information
	W				H					prior to command
	W				R					execution
	W				N					
	W				EOT					
	W				GPL					
	W				STP					
Execution										Data compared between the FDD and main-system
Result	R				ST 0					Status information after
	R				ST 1					command execution
	R				ST 2					
	R				C					Sector ID information
	R				H					
	R				R					
	R				N					
					SCAN LOW OR EQUAL					
Command	W	MT	MF	SK	1	1	0	0	1	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W				C					Sector ID information
	W				H					prior to command
	W				R					execution
	W				N					
	W				EOT					
	W				GPL					
	W				STP					
Execution										Data compared between the FDD and main-system
Result	R				ST 0					Status information after
	R				ST 1					command execution
	R				ST 2					
	R				C					Sector ID information
	R				H					after command execution
	R				R					
	R				N					

PHASE	R/W	D7	D6	D5	D4	D3	D2	D1	D0	REMARKS
					SCAN HIGH OR EQUAL					
Command	W	MT	MF	SK	1	1	1	0	1	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W				C					Sector ID information
	W				H					prior to command
	W				R					execution
	W				N					
	W				EOT					
	W				GPL					
	W				STP					
Execution										Data compared between the FDD and main-system
Result	R				ST 0					Status information after
	R				ST 1					command execution
	R				ST 2					
	R				C					Sector ID information
	R				H					after command execution
	R				R					
	R				N					
					RECALIBRATE					
Command	W	0	0	0	0	0	1	1	1	Command Codes
	W	X	X	X	X	X	0	US1	US0	
Execution No Result Phase										Head retracted to track 0
					SENSE INTERRUPT STATUS					
Command	W	0	0	0	0	1	0	0	0	Command Codes
Result	R				ST 0					Status information at
	R				PCN					the end of seek operation about the FDC
					SPECIFY					
Command	W	0	0	0	0	0	0	1	1	Command Codes
	W	—SRT—						—HUT—		
	W	—HLT—							ND	
No Result Phase										
					SENSE DRIVE STATUS					
Command	W	0	0	0	0	0	1	0	0	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
Result	R				ST 3					Status information about FDD
					SEEK					
Command	W	0	0	0	0	1	1	1	1	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W				NCN					
Execution										Head is positioned over proper cylinder on diskette
No Result Phase										
					INVALID					
Command	W				Invalid Codes					Invalid command codes (NoOp – FDC goes into standby state)
Result	R				ST 0					ST 0 = 80

Table 14. Status Register 0

BIT NO.	NAME	SYMBOL	DESCRIPTION
D7 D6	Interrupt Code	IC	D7 = 0 and D6 = 0 Normal termination of command, (NT), Command was completed and properly executed. D7 = 0 and D6 =1 Abnormal termination of command, (AT). Execution of command was started, but was not successfully completed. D7 =1 and D6 = 0 Invalid command issue (IC). Command which was issued was never started. D7 = 1 and D6 =1 Abnormal termination because during command execution the ready signal from FDD changed state.
D5	Seek End	SE	When the FDC completes the Seek command, this flag is set to 1 (high).
D4	Equipment Check	EC	If a fault signal is received from the FDD, or if the track 0 signal fails to occur after 77 step pulses (recalibrate command) then this flag is set.
D3	Not Ready	NR	When the FDD is in the not-ready state and a read or write command is issued, this flag is set. If a read or write command is issued to side 1 of a single sided drive, then this flag is set.
D2	Head Address	HD	This flag is used to indicate the state of the head at interrupt.
D1 D0	Unit Select 1 Unit Select 0	US 1 US 0	These flags are used to indicate a Drive unit Number at interrupt.

Table 15. Status Register 1

BIT NO.	NAME	SYMBOL	DESCRIPTION
D7	End of Cylinder	EN	When the FDC tries to access a sector beyond the final sector of a cylinder, this flag is set.
D6	–	–	Not used. This bit is always 0 (low).
D5	Data Error	DE	When the FDC detects a CRC error in either the ID field or the data field, this flag is set.
D4	Over Run	OR	If the FDC is not serviced by the main-systems during data transfers within a certain time interval, this flag is set.
D3	–	–	Not used. This bit is always 0 (low).
D2	No Data	ND	During Execution of a Read Data, Write Deleted Data, or Scan command, if the FDC cannot find the sector specified in the ID register, this flag is set. During execution of the Read ID command, if the FDC cannot read the ID field without an error, then this flag is set. During the execution of the Read-a-Cylinder command, if the starting sector cannot be found, then this flag is set.
D1	Not Writable	NW	During Execution of a Write Data, Write Deleted Data, or Format a Cylinder command, if the FDC detects a write protect signal from the FDD, then this flag is set.
D0	Missing Address Mark	MA	If the FDC cannot detect the ID Address Mark, this flag is set. Also at the same time, the MD (Missing Address Mark in Data Field) of Status Register 2 is set.

Table 16. Status Register 2

BIT NO.	NAME	SYMBOL	DESCRIPTION
D7	–	–	Not Used. This bit is always 0 (low).
D6	Control Mark	CM	During execution of the Read Data or Scan command, if the FDC encounters a sector which contains a Deleted Data Address Mark, this flag is set.
D5	Data Error in Data Field	DD	If the FDC detects a CRC error in the data then this flag is set.
D4	Wrong Cylinder	WC	This bit is related with the ND bit, and when the contents of C on the medium are different from that stored in the ID Register, this flag is set.
D3	Scan Equal Hit	SH	During execution of the Scan command, if the condition of "equal" is satisfied, this flag is set.
D2	Scan Not Satisifed	SN	During execution of the Scan command, if the FDC cannot find a sector on the cylinder which meets the condition, then this flag is set.
D1	Bad Cylinder	BC	This bit is related with the ND bit, and when the contents of C on the medium are different from that stored in the ID Register, and the content of C is FF, then this flag is set.
D0	Missing Address Mark in Data Field	MD	When data is read from the medium, if the FDC cannot find a Data Address Mark or Deleted Data Address Mark, then this flag is set.

Table 17. Status Register 3

BIT NO.	NAME	SYMBOL	DESCRIPTION
D7	Fault	FT	This bit is used to indicate the status of the Fault signal from the FDD.
D6	Write Protected	WP	This bit is used to indicate the status of the Write Protected signal from the FDD.
D5	Ready	RY	This bit is used to indicate the status of the Ready signal from the FDD.
D4	Track 0	T0	This bit is used to indicate the status of the Track 0 signal from the FDD.
D3	Two Side	TS	This bit is used to indicate the status of the Two Side signal from the FDD.
D2	Head Address	HD	This bit is used to indicate the status of Side Select signal to the FDD.
D1	Unit Select 1	US 1	This bit is used to indicate the status of the Unit Select 1 signal to the FDD.
D0	Unit Select 0	US 0	This bit is used to indicate the status of the Unit Select 0 signal to the FDD.

Programming Summary

DPC Registers (Ports)

FDC Data Reg — I/O Address 3F5

FDC Main Status Reg — I/O Address 3F4

Digital Output Reg — I/O Address 3F2

Bit 0	Drive	00: DR #A 10: DR #C
1	Select	01: DR #B 11: DR #D
2	Not FDC Reset	
3	Enable INT & DMA Requests	
4	Drive A Motor Enable	
5	Drive B Motor Enable	
6	Drive C Motor Enable	
7	Drive D Motor Enable	

All bits cleared with channel reset.

Interrupt 6

DMA 2

100 Disk Format

1 Head, 45 cylinders, 8 sectors/TRK, 512 bytes/sector,MFM.

FDC Constants

N: H'02', SC: 08, HUT: F, SRT: C, GPL FORMAT: H'05', GPL RD/WR: 2A, HLT: 01, (8ms track-track)

Drive Constants

HD Load	35 ms
HD Settle	25 ms
Motor Start	500 ms

Comments

1. Head loads with drive select, wait HD Load time before RD/WR.
2. Following access, wait HD Settle time before RD/WR.
3. Drive motors should be off when not in use. Only A or B and C or D may run simultaneously. Wait Motor Start time before RD/WR.
4. Motor must be on for drive to be selected.
5. Data Errors can occur while using a Home Television as the system display. Locating the TV too close to the diskette area can cause this to occur. To correct the problem, move the TV away from, or to the opposite side of the System Unit.

System I/O Channel Interface

All signals are TTL compatible:

MPUL 5.5 Vdc
LPUL 2.7 Vdc
MPDL 0.5 Vdc
LPDL –0.5 Vdc

The following lines are used by this adapter.

+D0–7 (Bidirectional, Load: 1 74LS; Driver: 74LS 3-state)

These eight lines form a bus by which all commands, status, and data are transferred. Bit 0 is the low-order bit.

+A0–9 (Adapter Input, Load: 1 74LS)

These ten lines form an address bus by which a register is selected to receive or supply the byte transferred via lines D0–7. Bit 0 is the low-order bit.

+AEN (Adapter Input, Load: 1 74LS)

The content of lines A0–9 is ignored if this line is active.

-IOW (Adapter Input, Load: 1 74LS)

The content of lines D0–7 is stored in the register addressed by lines A0–9 or DACK2 at the trailing edge of this signal.

-IOR (Adapter Input, Load: 1 74LS)

The content of the register addressed by lines A 0–9 or DACK2 is gated onto lines D0–7 when this line is active.

-DACK2 (Adapter Input, Load: 2 74LS)

This line active degates output DRQ2, selects the FDC data register as the source/destination of bus D0–7, and indirectly gates T/C to IRQ6.

+T/C (Adapter Input, Load: 4 74LS)

This line and DACK2 active indicates that the byte of data for which the DMA count was initialized is now being transferred.

+RESET (Adapter Input, Load: 1 74LS)

An up level aborts any operation in process and clears the Digital Output Register (DOR).

+DRQ2 (Adapter Output, Driver: 74LS 3-state)

This line is made active when the attachment is ready to transfer a byte of data to or from main storage. The line is made inactive by DACK2 becoming active or an I/O read of the FDC data register.

+IRQ6 (Adapter Output, Driver: 74LS 3-state)

This line is made active when the FDC has completed an operation. It results in an interrupt to a routine which should examine the FDC result bytes to reset the line and determine the ending condition.

Drive A & B Interface

All signals are TTL compatible:

MPUL 5.5 Vdc
LPUL 2.4 Vdc
MPDL 0.4 Vdc
LPDL –0.5 Vdc

All adapter outputs are driven by open-collector gates. The drive(s) must provide termination networks to Vcc (except Motor Enable 1 which has a 2K ohm resistor to Vcc).

Each adapter input is terminated with a 150 ohm resistor to Vcc.

Adapter Outputs

-Drive Select A&B (Driver: 7438)

These two lines are used by drives A&B to degate all drivers to the adapter and receivers from the attachment (except Motor Enable) when the line associated with a drive is inactive.

-Motor Enable A&B (Driver: 7438)

The drive associated with each of these lines must control its spindle motor such that it starts when the line becomes active and stops when the line becomes inactive.

-Step (Driver: 7438)

The selected drive moves the read/write head one cylinder in or out per the direction line for each pulse present on this line.

-Direction (Driver: 7438)

For each recognized pulse of the step line the read/write head moves one cylinder toward the spindle if this line is active, and away from the spindle if inactive.

-Head Select (Driver: 7438)

Head 1 (upper head) will be selected when this line is active (low).

-Write Data (Driver: 7438)

For each inactive to active transition of this line while Write Enable is active, the selected drive causes a flux change to be stored on the disk.

Adapter Inputs

-Write Enable (Driver: 7438)

The drive disables write current in the head unless this line is active.

-Index

The selected drive supplies one pulse per disk revolution on this line.

-Write Protect

The selected drive makes this line active if a write protected diskette is mounted in the drive.

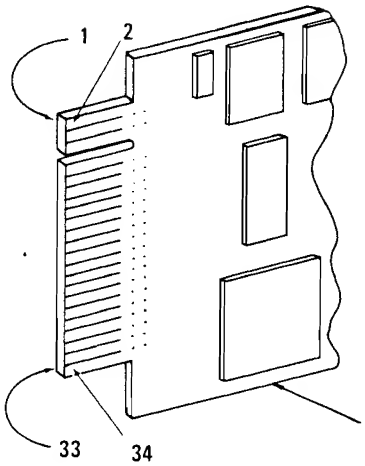
-Track 0

The selected drive makes this line active if the read/write head is over track 0.

-Read Data

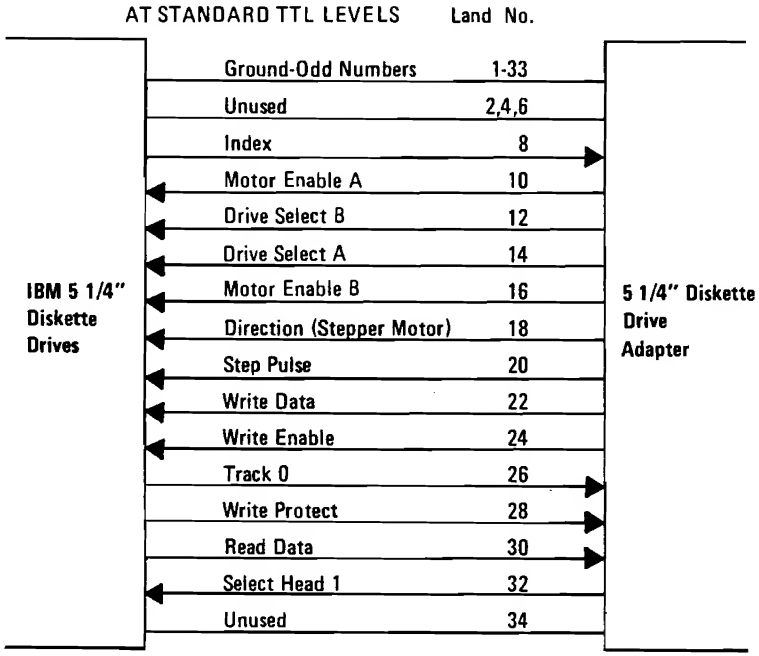
The selected drive supplies a pulse on this line for each flux change encountered on the disk.

5-1/4" Diskette Drive Adapter Internal Interface Specifications

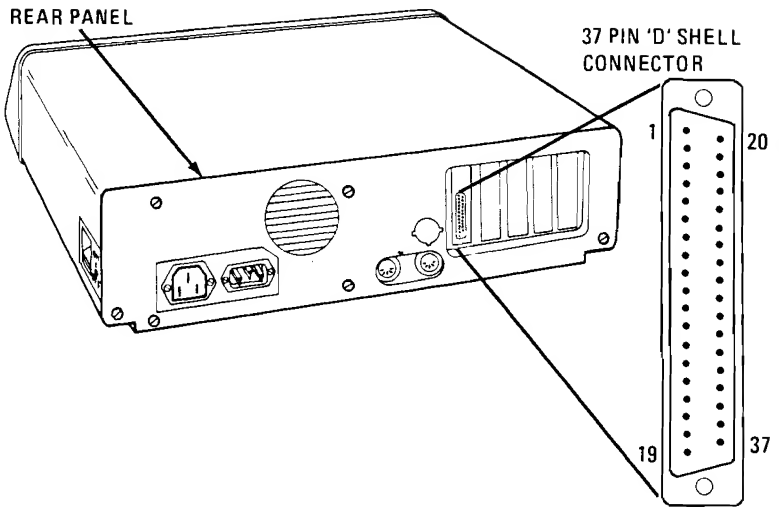


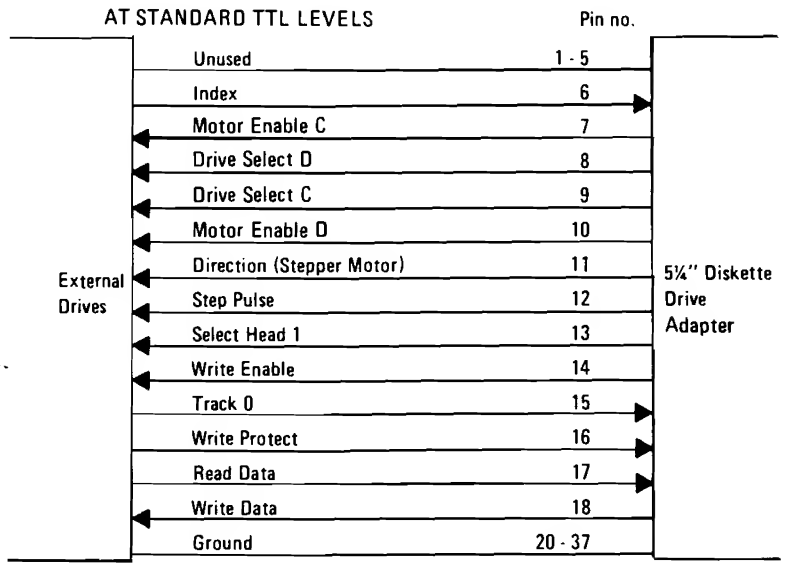
34 PIN KEYED
EDGE CONNECTOR

NOTE: LANDS 1–33 ARE ON THE BACKSIDE OF THE BOARD, LANDS 2–34 ARE ON THE FRONT, OR COMPONENT SIDE.



5-1/4" Diskette Drive Adapter External Interface Specifications





5-1/4" Diskette Drive

The IBM 5-1/4" Diskette Drive is a single or double sided, double density, 40 track per side unit. The Diskette Drive has a formatted capacity of 163,840 bytes for single sided and 327,680 bytes for double sided, and is capable of reading and recording digital data using Modified Frequency Modulation (MFM) methods. User access for diskette loading is provided by way of a slot located at the front of the unit.

The Diskette Drive is fully self-contained and requires no operator intervention during normal operation. The Drive consists of a spindle drive system, a head positioning system, and read/write/erase system.

When the front latch is opened, access is provided for the insertion of a diskette. The diskette is positioned in place by plastic guides, and the front latch. In/out location is ensured when the diskette is inserted until a back stop is encountered.

Closing the front latch activates the cone/clamp system resulting in centering of the diskette and clamping of the diskette to the drive hub. The drive hub is driven at a constant speed of 300 rpm by a servo controlled DC motor. In operation, the magnetic head is loaded into contact with the recording medium whenever the front latch is closed.

The magnetic head is positioned over the desired track by means of a 4-phase stepper motor/band assembly and its associated electronics. This positioner employs a one-step rotation to cause a 1-track linear movement. When a write-protected diskette is inserted into the Drive, the write-protect sensor disables the write electronics of the Drive and an appropriate signal is applied to the interface.

When performing a write operation, a 0.33 mm (0.013-in.) data track is recorded. This track is then tunnel erased to 0.30 mm (0.012-in).

Data recovery electronics include a low-level read amplifier, differentiator, zero-crossing detector, and digitizing circuits. All data decoding is provided by the adapter card.

The Drive is also supplied with the following sensor systems:

(1) A track 00 switch which senses when the Head/Carriage assembly is positioned at Track 00.

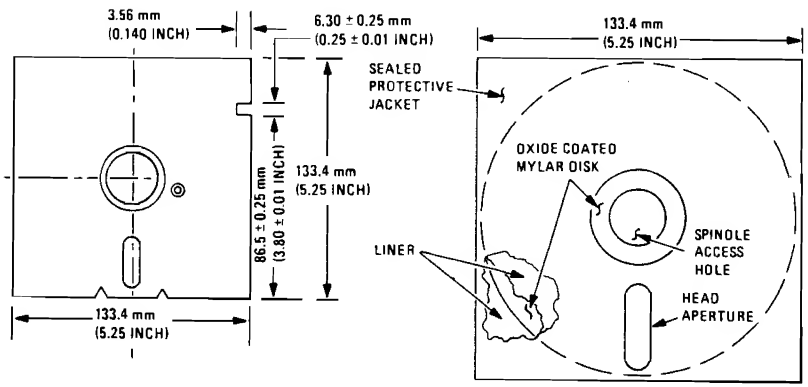
(2) The index sensor, which consists of a LED light source and phototransistor, is positioned so that when an index hole is detected, a digital signal is generated.

(3) The write-protect sensor disables the Diskette Drive electronics whenever a write-protect tab is applied to the diskette.

For interface information, refer to the Diskette Drive Adapter section.

Diskettes

The IBM 5-1/4" Diskette Drive uses a standard 133.4 mm (5.25 in.) diskette. For programming considerations, single sided, double density, soft sectored diskettes are used for single sided drives and double sided, double density, soft sectored diskettes are used for double sided drives. The figure below is a simplified drawing of the diskette used with the Diskette Drive. This recording medium is a flexible magnetic disk enclosed in a protective jacket. The protected disk, free to rotate within the jacket, is continuously cleaned by the soft fabric lining of the jacket during normal operation. Read/Write erase head access is made through an opening in the jacket. Openings for the drive hub and diskette index hole are also provided.



RECORDING MEDIUM

Table 18. Mechanical and Electrical Specifications

Media	Industry-compatible 5¼-inch diskette
Tracks per inch	48
Number of Tracks	(40)
Dimensions	
Height	85.85 mm (3.38 inches)
Width	149.10 mm (5.87 inches)
Depth	203.2 mm (8.0 inches)
Weight	2.04 Kg (4.5 lbs.)
Temperature	
(Exclusive of Media)	
Operating	10°C to 44°C (50°F to 112°F)
Non-operating	–40°C to 60°C (–40°F to 140°F)
Relative Humidity	
(Exclusive of Media)	
Operating	20% to 80% (Non-condensing)
Non-operating	5% to 95% (Non-condensing)
Seek Time	8 msec track to track
Head Setting Time	25 msec (last track addressed)
Error Rate	1 per 10^9 (recoverable) 1 per 10^{12} (non-recoverable) 1 per 10^6 (seeks)
Head Life	20,000 hours (normal use)
Media Life	3.0×10^6 passes per track
Disk Speed	300 rpm ± 1.5% (long term)
Instantaneous Speed Variation	± 3.0%
Start/Stop Time	500 msec (maximum)
Transfer Rate	250K bits/sec
Recording Mode	MFM
Power	+12 dc ± 0.6v 900 ma AVE. +5v dc ± 0.25 v, 600 ma AVE.

Memory Expansion Options

Three Memory Expansion Options are offered for the IBM Personal Computer. They are the 32KB, the 64KB, and the 64/256KB Memory Expansion Options. The base 64/256KB Option comes standard with 64KB of memory. One, two, or three 64KB Memory Module Kits may be added, providing the 64/256KB Option with 128KB, 192KB, or 256KB of memory respectively. The Memory Expansion Options plug into any of the five System Expansion Slots on the System Board and are used to extend system memory beyond 64KB. A maximum of 64KB of memory may be installed on the System Board as modules without using any System Expansion Slots or Expansion Options.

An expansion option must be configured to reside at a sequential 32K or 64K memory address boundary within the system address space. This is done by setting DIP switches on the option.

The expansion options are designed with 250 ns access dynamic memory chips. The 32KB and the 64KB options both use 16K x 1 memory chips while the 64/256KB option uses 64K x 1 memory chips. On the 32KB and the 64/256KB cards, 16 pin industry standard parts are used. On the 64KB card, stacked modules are used resulting in a 32K x 1, 18 pin module. This allows the 32KB and 64KB cards to have approximately the same physical dimensions.

All expansion options are parity checked and if a parity error is detected, a latch is set and an I/O channel check line is activated, indicating an error to the processor.

In addition to the memory modules, the expansion options contain the following circuits: bus buffering, dynamic memory timing generation, address multiplexing, and card select decode logic.

Dynamic memory refresh timing and address generation are functions which are not performed on the expansion options but are done once on the System Board and made available in the I/O channel for all devices.

To allow the System to address 32KB, 64KB, 64/256KB Memory Expansion Options, refer to the system configuration switch settings on page 2-28.

Operating Characteristics

The System Board operates at a frequency of 4.77 Mhz, which results in a clock frequency of 210 ns.

Normally, five clock cycles are required for a bus cycle so that a 1.05 μs memory cycle time is achieved. Memory write and memory read cycles both take five clock cycles, or 1.05 μs.

General specifications for memory used on all cards are:

Access - 250 ns
Cycle - 410 ns

Memory Module Description

Both the 32KB and 64KB options contain 18 dynamic memory modules. The 32KB Memory Expansion Option utilizes 16K x 1 bit modules and the 64KB Memory Expansion Option utilizes 32K x 1 bit modules.

The 64/256KB card has four banks of 9 pluggable connectors. Each bank will accept a 64KB Memory Module Kit, consisting of 9 (64K x 1) modules. The kits must be installed sequentially into Banks 1, 2, and 3. The base 64/256KB card comes standard with the first bank of modules installed into Bank 0, providing 64KB of memory. One, two, or three 64KB Kits may be added, upgrading the card to 128KB, 192KB, or 256KB of memory.

The 16K x 1 and the 32K x 1 modules require three voltage levels: +5 Vdc, -5 Vdc, and +12 Vdc. The 64K x 1 modules only require one voltage level of +5 Vdc. All three memory modules require 128 refresh cycles every 2 msec. Absolute maximum access times are:

From RAS: 250 ns
From CAS: 160 to 165 ns

Table 19. Memory Module Pin Configuration

PIN NO.	16K X 1 BIT MODULE (Used on 32KB Card)	32K X 1 BIT MODULE (Used on 64KB Card)	64K X 1 BIT MODULE (Used on 64/256KB Card)
1	−5V	−5V	N/C
2	Data In**	Data In**	Data In***
3	−Write	−Write	−Write
4	−RAS	−RAS 0	−RAS
5	A0	−RAS 1	A0
6	A2	A0	A2
7	A1	A2	A1
8	+12V	A1	+5VDC
9	+5V	+12V	A7
10	A5	+5V	A5
11	A4	A5	A4
12	A3	A4	A3
13	A6	A3	A6
14	Data Out**	A6	Data Out***
15	−CAS	Data Out**	−CAS
16	GND	−CAS 1	GND
17	−*	−CAS 0	*
18	−*	GND	*

* 16K X 1 and 64K X 1 bit module has only 16 pins.
** Data In and Data Out are tied together (three state bus).
*** Data In and Data Out are tied together on Data Bits 0−7 (three state bus).

Switch-Configurable Start Address

Each card has a small DIP Module which contains eight switches. The switches are used to set the card start address as follows:

Table 20. DIP Module Start Address

NO.	32KB AND 64KB OPTIONS	64/256KB OPTION
1	ON: A19=0; OFF: A19=1	ON: A19=0; OFF: A19=1
2	ON: A18=0; OFF: A18=1	ON: A18=0; OFF: A18=1
3	ON: A17=0; OFF: A17=1	ON: A17=0; OFF: A17=1
4	ON: A16=0; OFF: A16=1	ON: A16=0; OFF: A16=1
5	ON: A15=0; OFF: A15=1*	ON: Select 64KB
6	Not Used	ON: Select 128KB
7	Not Used	ON: Select 192KB
8	Used Only In 64KB RAM Card*	ON: Select 256KB

* Switch No. 8 may be set on the 64KB Memory Expansion. Option to use only half the memory on the card (i.e., 32KB). If Switch No. 8 is ON, all 64KB is accessible. If Switch No. 8 is OFF, address bit A15 (as set by Switch No. 5) is used to determine which 32KB are accessible and the 64KB option behaves exactly like a 32KB option.

NOTES

Game Control Adapter

The Game Control Adapter allows the system to attach paddles and joysticks. Up to four paddles or two joysticks may be attached. In addition, four inputs for switches are provided. Paddle and joystick positions are determined by changing resistive values sent to the adapter. The adapter plus system software converts the present resistive value to a relative paddle or joystick position. On receipt of an output signal, four timing circuits are started. By determining the time required for the circuit to time out (a function of the resistance), the paddle position can be determined. This card could be used as a general purpose I/O card with four analog (resistive) inputs plus four digital input points. This card fits into any of the five System Board I/O slots. The game control interface cable attaches to the rear of the card which protrudes through the rear panel of the System Unit.

Game Control Adapter Block Diagram

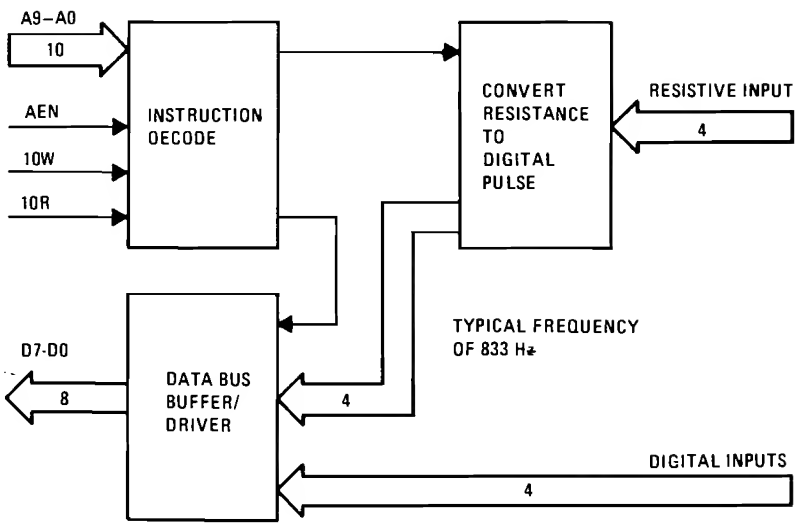


Figure 20. GAME CONTROL ADAPTER BLOCK DIAGRAM

Functional Description

Address Decode

The select on the Game Control Adapter is generated by two 74LS138's as an address decoder. AEN must be inactive while the address is 201 in order to generate the select. The select allows a write to fire the one-shots or a read to give the values of the trigger buttons and one-shot outputs.

Data Bus Buffer/Driver

The data bus is buffered by a 74LS244 buffer/driver. For an IN from address X'201', the Game Control Adapter will drive the data bus; at all other times the buffer is left in the high impedance state.

Trigger Buttons

The trigger button inputs are read via an IN from address X'201'. A trigger button is on each joystick/paddle. These values are seen on data bits 7 through 4 (see Software Interface sub-section). These buttons default to an open state and are read as "1". When a button is depressed, it is read as "0". Software should be aware that these buttons are NOT debounced in hardware.

Joystick Positions

The joystick position is indicated by a potentiometer for each coordinate. Each potentiometer has a range from 0 to 100K ohms that varies the time constant for each of the four one-shots. As this time constant is set at different values, the output of the one-shot will be of varying durations.

All four one-shots are fired at once by an OUT to address X'201'. All four one-shot outputs will go true after the fire pulse and will remain high for varying times depending on where each potentiometer is set.

These four one-shot outputs are read via an IN from address X'201' and are seen on data bits 3 through 0.

I/O Channel Description

A9-A0:	Address lines 9 through 0 are used to address the Game Control Adapter.
D7-D0:	Data lines 7 through 0 are the data bus.
IOR, IOW:	I/O Read and I/O Write are used when reading from or writing to an adapter (IN, OUT).
AEN:	When active, the adapter must be inactive and the data bus driver inactive.
+5V:	Power for the Game Control Adapter.
GND:	Common ground.
A19-A10:	Unused
MEMR, MEMW:	Unused
DACK0-DACK3:	Unused
IRQ7-IRQ2:	Unused
DRQ3-DRQ1:	Unused
ALE, T/C:	Unused
CLK, OSC:	Unused
I/O CHCK:	Unused
I/O CH RDY:	Unused
HRQ I/O CH:	Unused
RESET DRV:	Unused
-5V, +12V, -12V:	Unused

Interface Description

The Game Control Adapter has 8 input lines, 4 of which are digital inputs and 4 of which are resistive inputs. The inputs are read with one IN from address x'201'.

The 4 digital inputs each have a 1K ohm pullup resistor to +5V. With no drive on these inputs, a '1' is read. For a '0' reading, the inputs must be pulled to ground.

The 4 resistive inputs, measured to +5V, will be converted to a digital pulse with a duration proportional to the resistive load, according to the following equation:

Time = 24.2 μsec + 0.011 (r) μsec

The user must first begin the conversion by an OUT to address x'201'. An IN from address x'201' will show the digital pulse go high and remain high for the duration according to the resistance value. All four bits (Bit 3-Bit 0) function in the same manner, their digital pulse will all go high simultaneously and will reset independently according to the input resistance value.

Input from address x'201'

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Digital Inputs				Resistive Inputs			

The typical input to the Game Control Adapter is a set of joysticks or game paddles.

The joysticks will typically have a set of two joysticks (A&B). These will have one or two buttons each with two variable resistances each, with a range from 0 to 100K ohms. One variable resistance will indicate the X coordinate and the other variable resistance will indicate the Y coordinate. This should be attached to give the following input data:

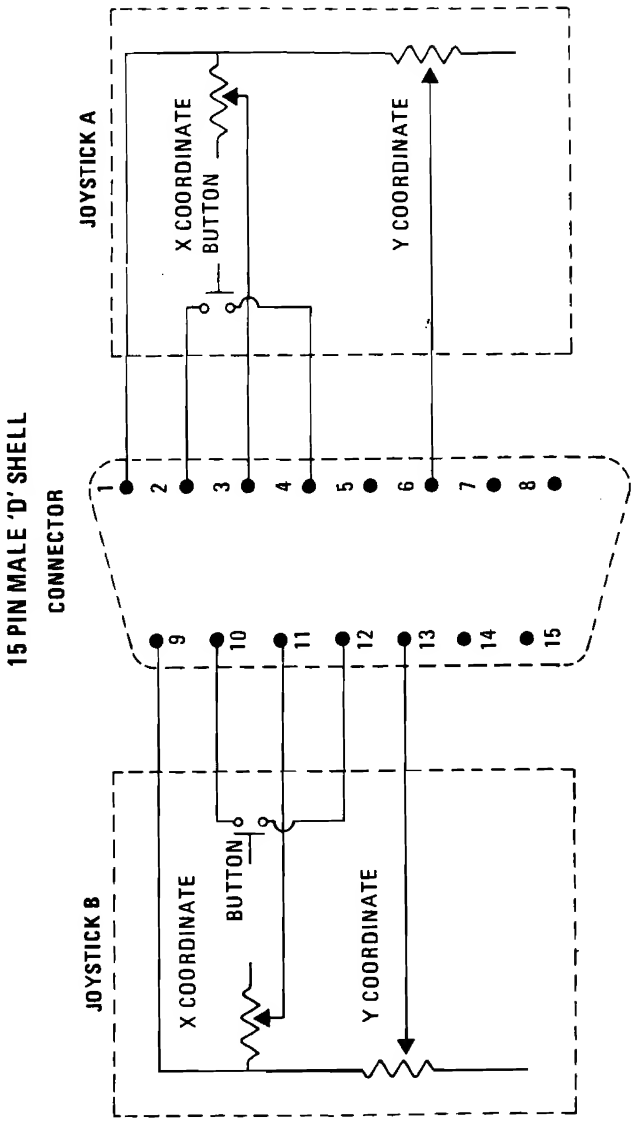
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
B-#2 Button	B-#1 Button	A-#2 Button	A-#1 Button	B-Y Coord.	B-X Coord.	A-Y Coord.	A-X Coord.

The game paddles will have a set of two (A&B) or four (A,B,C, & D) paddles. These will have one button each and one variable resistance each, with a range from 0 to 100K ohms. This should be attached to give the following input data:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
D Button	C Button	B Button	A Button	D Coord.	C Coord.	B Coord.	A Coord.

A schematic diagram for attaching a set of game controllers is on page 2-135.

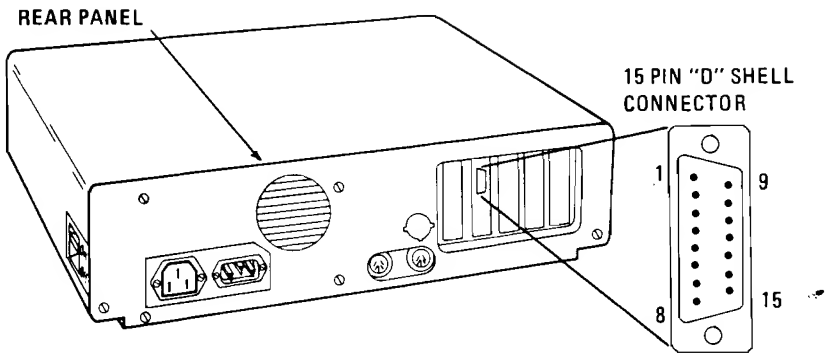
Joystick Schematic

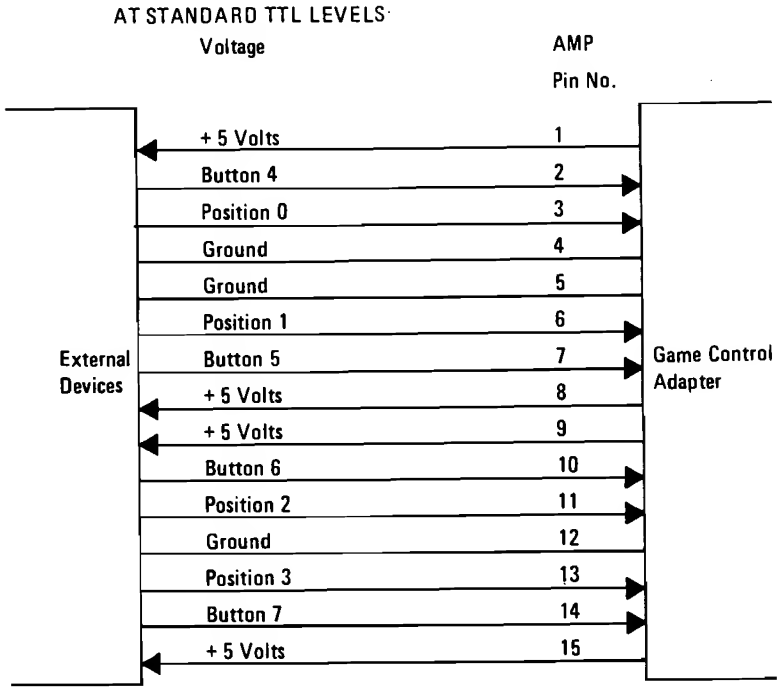


NOTE: POTENTIOMETER FOR X & Y COORDINATES HAS A RANGE OF 0 TO 100KΩ.
BUTTON IS NORMALLY OPEN; CLOSED WHEN DEPRESSED.

Figure 21. JOYSTICK SCHEMATIC

Game Controller Adapter (Analog Input) Connector Specifications





Asynchronous Communications Adapter

The Asynchronous Communications Adapter is a 4"H x 5"W card that plugs into a System Expansion Slot. All system control signals and voltage requirements are provided through a 2 x 31 position card edge tab. A jumper module is provided to select either RS-232-C or current loop operation.

The adapter is fully programmable and supports asynchronous communications only. It will add and remove start bits, stop bits, and parity bits. A programmable baud rate generator allows operation from 50 baud to 9600 baud. Five, six, seven or eight bit characters with 1, 1-1/2, or 2 stop bits are supported. A fully prioritized interrupt system controls transmit, receive, error, line status and data set interrupts. Diagnostic capabilities provide loopback functions of transmit/receive and input/output signals.

Figure (22) is a block diagram of the Asynchronous Communications Adapter.

The heart of the adapter is a INS8250 LSI chip or functional equivalent. The following is a summary of the 8250's key features:

- Adds or Deletes Standard Asynchronous Communication Bits (Start, Stop, and Parity) to or from Serial Data Stream.
- Full Double Buffering Eliminates Need for Precise Synchronization.
- Independently Controlled Transmit, Receive, Line Status, and Data Set Interrupts.
- Programmable Baud Rate Generator Allows Division of Any Input Clock by 1 to $(2^{16}-1)$ and Generates the Internal 16x Clock.
- Independent Receiver Clock Input.
- MODEM Control Functions – Clear to Send (CTS), Request to Send (RTS), Data Set Ready (DSR), Data Terminal Ready (DTR), Ring Indicator (RI), and Carrier Detect (CD).
- Fully Programmable Serial-Interface Characteristics
  - 5-, 6-, 7-, or 8-Bit Characters
  - Even, Odd, or No-Parity Bit Generation and Detection
  - 1-, 1-1/2, or 2-Stop Bit Generation
  - Baud Rate Generation (DC to 9600 Baud)

- False Start Bit Detection.
- Complete Status Reporting Capabilities.
- Line Break Generation and Detection.
- Internal Diagnostic Capabilities.
  - Loopback Controls for Communications Link Fault Isolation.
  - Break, Parity, Overrun, Framing Error Simulation.
- Full Prioritized Interrupt System Controls.

All communications protocol is a function of the system microcode and must be loaded before the adapter is operational. All pacing of the interface and control signal status must be handled by the system software.

Asynchronous Communications Block Diagram

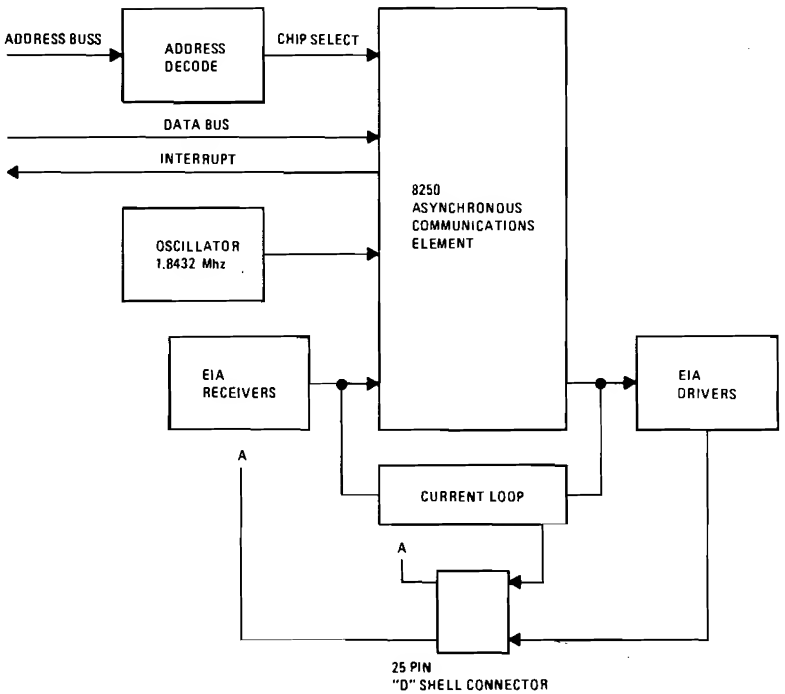


Figure 22. ASYNCHRONOUS COMMUNICATIONS ADAPTER BLOCK DIAGRAM

Modes of Operation

The different modes of operation are selected by programming the 8250 Asynchronous Communications Element. This is done by selecting the I/O address (3F8 to 3FF) and writing data out to the card. Address bit A0, A1 and A2 select the different registers which define the modes of operation. Also, the Divisor Latch Access Bit (Bit 7) of the line control register is used to select certain registers.

I/O Decode for Communications Adapter

Table 21. I/O Decodes (3F8 - 3FF)

I/O DECODE	REGISTER SELECTED	DLAB STATE
3F8	TX BUFFER	DLAB=0 (WRITE)
3F8	RX BUFFER	DLAB=0 (READ)
3F8	DIVISOR LATCH LSB	DLAB=1
3F9	DIVISDR LATCH MSB	DLAB=1
3F9	INTERRUPT ENABLE REGISTER	DLAB=0
3FA	INTERRUPT IDENTIFICATION REGISTERS	
3FB	LINE CONTROL REGISTER	
3FC	MODEM CONTROL REGISTER	
3FD	LINE STATUS REGISTER	
3FE	MODEM STATUS REGISTER	

ADDRESS BITS

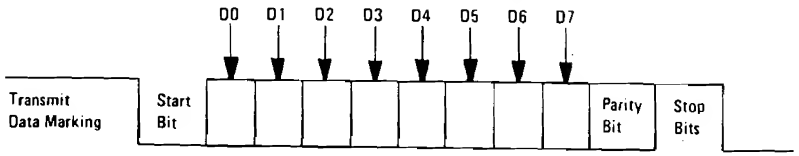
	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	DLAB	REGISTER
3F8 to 3FF	1	1	1	1	1	1	1	X	X	X		
								0	0	0	0	Receive Buffer (read), Transmit Holding Reg. (write)
								0	0	1	0	Interrupt Enable
								0	1	0	X	Interrupt Identification
								0	1	1	X	Line Control
								1	0	0	X	Modem Control
								1	0	1	X	Line Status
								1	1	0	X	Modem Status
								1	1	1	X	None
								0	0	0	1	Divisor Latch (LSB)
								0	0	1	1	Divisor Latch (MSB)

A2, A1 and A0 bits are "Don't Cares" and are used to select the different register of the communications chip.

One interrupt line is provided to the system. This interrupt is IRQ4 and will be positive active. To allow the communications card to send interrupts to the system, Bit 3 of the Modem Control Register must be set = 1 (high). At this point, any interrupts allowed by the Interrupt Enable Register will cause an interrupt.

The data format will be as follows:

TRANSMITTER OUTPUT AND RECEIVER INPUT



Data Bit 0 is the first bit to be transmitted or received. The adapter automatically inserts the start bit, the correct parity bit if programmed to do so, and the stop bit (1, 1-1/2 or 2 depending on the command in the Line Control Register).

Interface Description

The communications adapter provides an EIA RS-232-C, or equivalent, interface. One 25 pin "D" shell, male type connector is provided to attach various peripheral devices. In addition, a current loop interface is also located in this same connector. A jumper block is provided to manually select either the voltage interface, or the current loop interface.

Pin 18 + receive current loop data (20Ma)
Pin 25 – receive current loop return (20Ma)
Pin 9 + transmit current loop return (20Ma)
Pin 11 - transmit current loop data (20Ma)

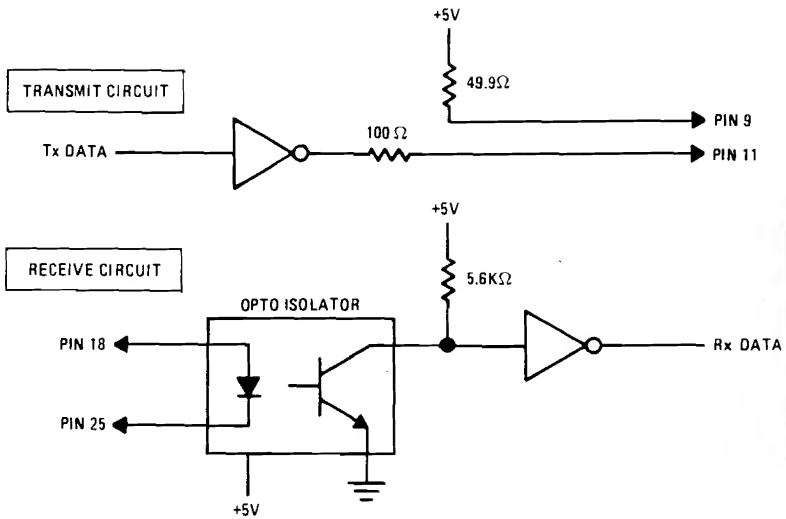


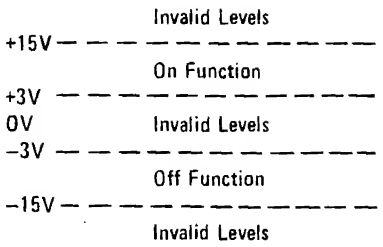
Figure 23. CURRENT LOOP INTERFACE

The voltage interface is a serial interface. It supports certain data and control signals as listed below:

Pin 2	Transmit Data
Pin 3	Receive Data
Pin 4	Request to Send
Pin 5	Clear to Send
Pin 6	Data Set Ready
Pin 7	Signal Ground
Pin 8	Carrier Detect
Pin 20	Data Terminal Ready
Pin 22	Ring Indicate

The adapter converts these signals from TTL to EIA voltage levels and from EIA to TTL voltage levels. These signals are sampled or generated by the communication control chip. These signals can then be sensed by the system software to determine the state of the interface or peripheral device.

Interchange Voltage	Binary State	Signal Condition	Interface Control Function
Positive Voltage =	Binary (0)	= Spacing	=On
Negative Voltage =	Binary (1)	= Marking	=Off



The signal will be considered in the "marking" condition when the voltage on the interchange circuit, measured at the interface point, is more negative than minus three volts with respect to signal ground. The signal will be considered in the "spacing" condition when the voltage is more positive than plus three volts with respect to signal ground. The region between plus three volts and minus three volts is defined as the transition region and is considered an invalid level. The voltage which is more negative than -15V or more positive than +15V will be considered in invalid levels.

During the transmission of data, the "marking" condition will be used to denote the binary state "one" and "spacing" condition will be used to denote the binary state "zero".

For interface control circuits, the function is "on" when the voltage is more positive than +3V with respect to signal ground and is "off" when the voltage is more negative than -3V with respect to signal ground.

INS8250 Functional Pin Description

The following describes the function of all INS8250 input/output pins. Some of these descriptions reference internal circuits.

Note: In the following descriptions, a low represents a logic 0 (0 volt nominal) and a high represents a logic 1 (+2.4 volts nominal).

Input Signals

Chip Select (CS0, CS1, CS2), Pins 12-14: When CS0 and CS1 are high and CS2 is low, the chip is selected. Chip selection is complete when the decoded chip select signal is latched with an active (low) Address Strobe (ADS) input. This enable communication between the INS8250 and the CPU.

Data Input Strobe (DISTR,$\overline{\text{DISTR}}$) Pins 22 and 21: When DISTR is high or DISTR is low while the chip is selected, allows the CPU to read status information or data from a selected register of the INS8250.

Note: Only an active DISTR or $\overline{\text{DISTR}}$ input is required to transfer data from the INS8250 during a read operation. Therefore, tie either the DISTR input permanently low or the DISTR input permanently high, if not used.

Data Output Strobe (DOSTR,$\overline{\text{DOSTR}}$), Pins 19 and 18: When DOSTR is high or DOSTR is low while the chip is selected, allows the CPU to write data or control words into a selected register of the INS8250.

Note: Only an active DOSTR or $\overline{\text{DOSTR}}$ input is required to transfer data to the INS8250 during a write operation. Therefore, tie either the DOSTR input permanently low or the DOSTR input permanently high, if not used.

Address Strobe (ADS), Pin 25: When low, provides latching for the Register Select (A0, A1, A2) and Chip Select (CS0, CS1, CS2) signals.

Note: An active ADS input is required when the Register Select (A0, A1, A2) signals are not stable for the duration of a read or write operation. If not required, tie the ADS input permanently low.

Register Select (A0, A1, A2), Pins 26-28: These three inputs are used during a read or write operation to select an INS8250 register to read from or write into as indicated in the table below. Note that the state of the Divisor Latch Access Bit (DLAB), which is the most significant bit of the Line Control Register, affects the selection of certain INS8250 registers. The DLAB must be set high by the system software to access the Baud Generator Divisor Latches.

DLAB	A2	A1	A0	Register
0	0	0	0	Receiver Buffer (read), Transmitter Holding Register (write)
0	0	0	1	Interrupt Enable
X	0	1	0	Interrupt Identification (read only)
X	0	1	1	Line Control
X	1	0	0	MODEM Control
X	1	0	1	Line Status
X	1	1	0	MODEM Status
X	1	1	1	None
1	0	0	0	Divisor Latch (least significant byte)
1	0	0	1	Divisor Latch (most significant byte)

Master Reset (MR), Pin 35: When high, clears all the registers (except the Receiver Buffer, Transmitter Holding, and Divisor Latches), and the control logic of the INS8250. Also, the state of various output signals (SOUT, INTRPT, OUT 1, OUT 2, RTS, DTR) are affected by an active MR input. (Refer to Table 1.)

Receiver Clock (RCLK), Pin 9: This input is the 16x baud rate clock for the receiver section of the chip.

Serial Input (SIN), Pin 10: Serial data input from the communications link (peripheral device, MODEM, or data set).

Clear to Send (CTS), Pin 36: The CTS signal is a MODEM control function input whose condition can be tested by the CPU by reading Bit 4 (CTS) of the MODEM Status Register. Bit 0 (DCTS) of the MODEM Status Register indicates whether the CTS input has changed state since the previous reading of the MODEM Status Register.

Note: Whenever the CTS bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.

Data Set Ready (DSR), Pin 37: When low, indicates that the MODEM or data set is ready to establish the communications link and transfer data with the INS8250. The DSR signal is a MODEM-control function input whose condition can be tested by the CPU by reading Bit 5 (DSR) of the MODEM Status Register. Bit 1 (DDSR) of the MODEM Status Register indicates whether the DSR input has changed state since the previous reading of the MODEM Status Register.

Note: Whenever the DSR bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.

Received Line Signal Detect (RLSD), Pin 38: When low, indicates that the data carrier has been detected by the MODEM or data set. The RLSD signal is a MODEM-Control function input whose condition can be tested by the CPU by reading Bit 7 (RLSD) of the MODEM Status Register. Bit 3 (DRLSD) of the MODEM Status Register indicates whether the RLSD input has changed state since the previous reading of the MODEM Status Register.

Note: Whenever the RLSD bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.

Ring Indicator (RI), Pin 39: When low, indicates that a telephone ringing signal has been received by the MODEM or data set. The RI signal is a MODEM-control function input whose condition can be tested by the CPU by reading Bit 6 (RI) of the MODEM Status Register. Bit 2 (TERI) of the MODEM Status Register indicates whether the RI input has changed from a low to a high state since the previous reading of the MODEM Status Register.

Note: Whenever the RI bit of the MODEM Status Register changes from a high to a low state, an interrupt is generated if the MODEM Status Interrupt is enabled.

VCC, Pin 40: +5 volt supply.

VSS, Pin 20: Ground (0-volt) reference.

Data Terminal Ready (DTR), Pin 33: When low, informs the MODEM or data set that the INS8250 is ready to communicate. The DTR output signal can be set to an active low by programming Bit 0 (DTR) of the MODEM Control Register to a high level. The DTR signal is set high upon a Master Reset operation.

Request to Send (RTS), Pin 32: When low, informs the MODEM or data set that the INS8250 is ready to transmit data. The RTS output signal can be set to an active low by programming Bit 1 (RTS) of the MODEM Control Register. The RTS signal is set high upon a Master Reset operation.

Output 1 (OUT 1), Pin 34: User-designated output that can be set to an active low by programming Bit 2 (OUT 1) of the MODEM Control Register to a high level. The OUT 1 signal is set high upon a Master Reset operation.

Output 2 (OUT 2), Pin 31: User-designated output that can be set to an active low by programming Bit 3 (OUT 2) of the MODEM Control Register to a high level. The OUT 2 signal is set high upon a Master Reset operation.

Chip Select Out (CSOUT), Pin 24: When high, indicates that the chip has been selected by active CS0, CS1, and CS2 inputs. No data transfer can be initiated until the CSOUT signal is a logic 1.

Driver Disable (DDIS), Pin 23: Goes low whenever the CPU is reading data from the INS8250. A high-level DDIS output can be used to disable an external transceiver (if used between the CPU and INS8250 on the D7-D0 Data Bus) at all times, except when the CPU is reading data.

Baud Out (BAUDOUT), Pin 15: 16x clock signal for the transmitter section of the INS8250. The clock rate is equal to the main reference oscillator frequency divided by the specified divisor in the Baud Generator Divisor Latches. The BAUDOUT may also be used for the receiver section by typing this output to the RCLK input of the chip.

Interrupt (INTRPT), Pin 30: Goes high whenever any one of the following interrupt types has an active high condition and is enabled via the IER: Receiver Error Flag; Received Data Available; Transmitter Holding Register Empty; and MODEM Status. The INTRPT Signal is reset low upon the appropriate interrupt service or a Master Reset operation.

Serial Output (SOUT), Pin 11: Composite serial data output to the communications link (peripheral, MODEM or data set). The SOUT signal is set to the Marking (Logic 1) state upon a Master Reset operation.

Data (D7-D0) Bus, Pins 1-8: This bus comprises eight TRI-STATE input/output lines. The bus provides bidirectional communications between the INS8250 and the CPU. Data, control words, and status information are transferred via the D7-D0 Data Bus.

External Clock Input/Output (XTAL1, XTAL2, Pins 16 and 17: These two pins connect the main timing reference (crystal or signal clock) to the INS8250.

Programming Considerations

Table 22. Asynchronous Communications Reset Functions

Register/Signal	Reset Control	Reset State
Interrupt Enable Register	Master Reset	All Bits Low (0–3 Forced and 4–7 Permanent)
Interrupt Identification Register	Master Reset	Bit 0 is High, Bits 1 and 2 Low Bits 3–7 are Permanently Low
Line Control Register	Master Reset	All Bits Low
MODEM Control Register	Master Reset	All Bits Low
Line Status Register	Master Reset	Except Bits 5 & 6 are High
MODEM Status Register	Master Reset	Bits 0–3 Low Bits 4–7 – Input Signal
SOUT	Master Reset	High
INTRPT (RCVR Errs)	Read LSR/MR	Low
INTRPT (RCVR Data Ready)	Read RBR/MR	Low
INTRPT (RCVR Data Ready)	Read IIR/Write THR/MR	Low
INTRPT (MODEM Status Changes)	Read MSR/MR	Low
OUT 2	Master Reset	High
RTS	Master Reset	High
DTR	Master Reset	High
OUT 1	Master Reset	High

INS8250 Accessible Registers

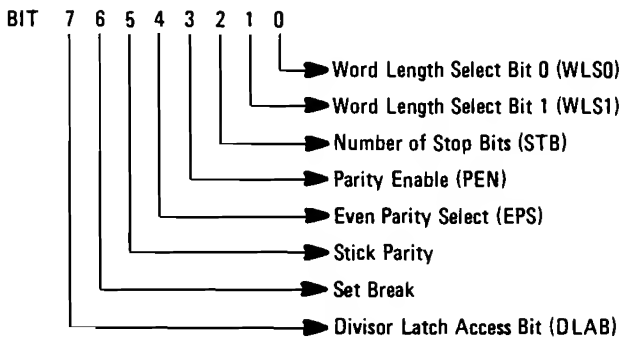
The system programmer may access or control any of the INS8250 registers via the CPU. These registers are used to control INS8250 operations and to transmit and receive data.

INS8250 Line Control Register

The system programmer specifies the format of the asynchronous data communications exchange via the Line Control Register. In addition to controlling the format, the programmer may retrieve the contents of the Line Control Register for inspection. This feature simplifies system programming and eliminates the need for separate storage in system memory of the line characteristics. The contents of the Line Control Register are indicated and described below.

Line Control Register (LCR)

3FB



Bit 0 and 1: These two bits specify the number of bits in each transmitted or received serial character. The encoding of bits 0 and 1 is as follows:

Bit 1	Bit 0	Word Length
0	0	5 Bits
0	1	6 Bits
1	0	7 Bits
1	1	8 Bits

Bit 2: This bit specifies the number of Stop bits in each transmitted or received serial character. If bit 2 is a logic 0, 1 Stop bit is generated or checked in the transmit or receive data, respectively. If bit 2 is logic 1 when a 5-bit word length is selected via bits 0 and 1, 1-1/2 Stop bits are generated or checked. If bit 2 is logic 1 when either a 6-, 7-, or 8-bit word length is selected, 2 Stop bits are generated or checked.

Bit 3: This bit is the Parity Enable bit. When bit 3 is a logic 1, a Parity bit is generated (transmit data) or checked (receive data) between the last data word bit and Stop bit of the serial data. (The Parity bit is used to produce an even or odd number of 1's when the data word bits and the Parity bit are summed.)

Bit 4: This bit is the Even Parity Select bit. When bit 3 is a logic 1 and bit 4 is a logic 0, an odd number of logic 1's is transmitted or checked in the data word bits and Parity bit. When bit 3 is a logic 1 and bit 4 is a logic 1, an even number of bits is transmitted or checked.

Bit 5: This bit is the Stick Parity bit. When bit 3 is a logic 1 and bit 5 is a logic 1, the Parity bit is transmitted and then detected by the receiver as a logic 0 if bit 4 is a logic 1 or as a logic 1 if bit 4 is a logic 0.

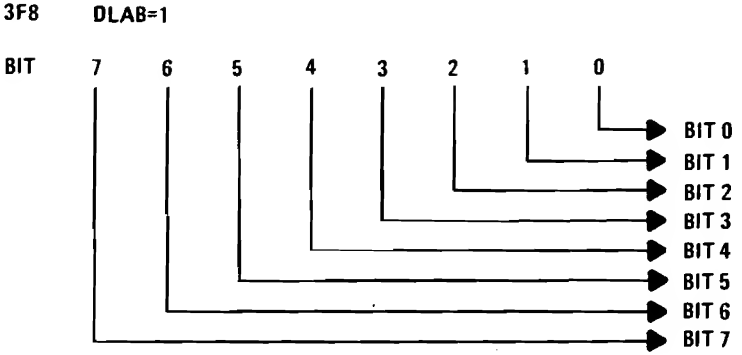
Bit 6: This bit is the Set Break Control bit. When bit 6 is a logic 1, the serial output (SOUT) is forced to the Spacing (logic 0) state and remains there regardless of other transmitter activity. The set break is disabled by setting bit 6 to a logic 0. This feature enables the CPU to alert a terminal in a computer communications system.

Bit 7: This bit is the Divisor Latch Access Bit (DLAB). It must be set high (logic 1) to access the Divisor Latches of the Baud Rate Generator during a Read or Write operation. It must be set low (logic 0) to access the Receiver Buffer, the Transmitter Holding Register, or the Interrupt Enable Register.

INS8250 Programmable Baud Rate Generator

The INS8250 contains a programmable Baud Rate Generator that is capable of taking the clock input (1.8432 Mhz) and dividing it by any divisor from 1 to $(2^{16}-1)$. The output frequency of the Baud Generator is 16x the Baud rate [divisor #= (frequency input)/(baud rate x 16)]. Two 8-bit latches store the divisor in a 16-bit binary format. These Divisor Latches must be loaded during initialization in order to ensure desired operation of the Baud Rate Generator. Upon loading either of the Divisor Latches, a 16-bit Baud counter is immediately loaded. This prevents long counts on initial load.

Divisor Latch Least Significant Bit (DLL)



Divisor Latch Most Significant Bit (DLM)

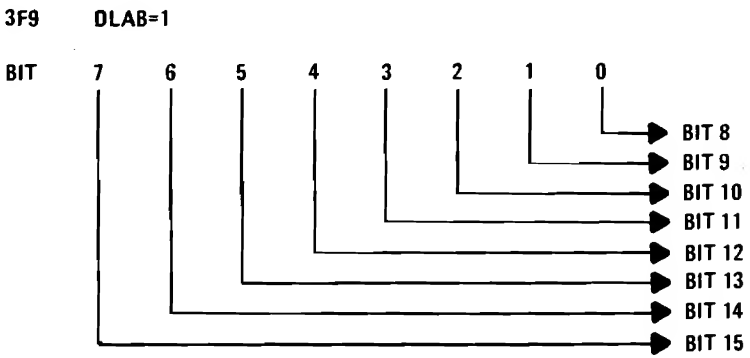


Table 23 illustrates the use of the Baud Rate Generator with a frequency of 1.8432 Mhz. For baud rates of 9600 and below, the error obtained is minimal.

Note: The maximum operating frequency of the Baud Generator is 3.1 Mhz. In no case should the data rate be greater than 9600 Baud.

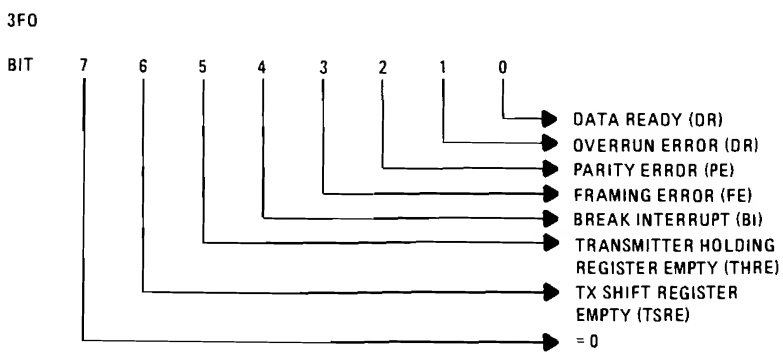
Table 23. BAUD Rate at 1.843 Mhz

Desired Baud Rate	Divisor Used to Generate 16x Clock		Percent Error Difference Between Desired & Actual
	Decimal	Hex	
50	2304	'900'	--
75	1536	'600'	--
110	1047	'417'	0.026
134.5	857	'359'	0.058
150	768	'300'	--
300	384	'180'	--
600	192	'0C0'	--
1200	96	'060'	--
1800	64	'040'	--
2000	58	'03A'	0.69
2400	48	'030'	--
3600	32	'020'	--
4800	24	'018'	--
7200	16	'010'	--
9600	12	'00C'	--

Line Status Register

This 8-bit register provides status information to the CPU concerning the data transfer. The contents of the Line Status Register are indicated and described below.

Line Status Register (LSR)



Bit 0: This bit is the receiver Data Ready (DR) indicator. Bit 0 is set to a logic 1 whenever a complete incoming character has been received and transferred into the Receiver Buffer Register. Bit 0 may be reset to a logic 0 either by the CPU reading the data in the Receiver Buffer Register or by writing a logic 0 into it from the CPU.

Bit 1: This bit is the Overrun Error (OE) indicator. Bit 1 indicates that data in the Receiver Buffer Register was not read by the CPU before the next character was transferred into the Receiver Buffer Register, thereby destroying the previous character. The OE indicator is reset whenever the CPU reads the contents of the Line Status Register.

Bit 2: This bit is the Parity Error (PE) indicator. Bit 2 indicates that the received data character does not have the correct even or odd parity, as selected by the even parity-select bit. The PE bit is set to a logic 1 upon detection of a parity error and is reset to a logic 0 whenever the CPU reads the contents of the Line Status Register.

Bit 3: This bit is the Framing Error (FE) indicator. Bit 3 indicates that the received character did not have a valid Stop bit. Bit 3 is set to a logic 1 whenever the Stop bit following the last data bit or parity bit is detected as a zero bit (Spacing level).

Bit 4: This bit is the Break Interrupt (BI) indicator. Bit 4 is set to a logic 1 whenever the received data input is held in the Spacing (logic 0) state for longer than a full word transmission time (that is, the total time of Start bit + data bits + Parity + Stop bits).

Note: Bits 1 through 4 are the error conditions that produce a Receiver Line Status interrupt whenever any of the corresponding conditions are detected.

Bit 5: This bit is the Transmitter Holding Register Empty (THRE) indicator. Bit 5 indicates that the INS8250 is ready to accept a new character for transmission. In addition, this bit causes the INS8250 to issue an interrupt to the CPU when the Transmit Holding Register Empty Interrupt enable is set high. The THRE bit is set to a logic 1 when a character is transferred from the Transmitter Holding Register into the Transmitter Shift Register. The bit is reset to logic 0 concurrently with the loading of the Transmitter Holding Register by the CPU.

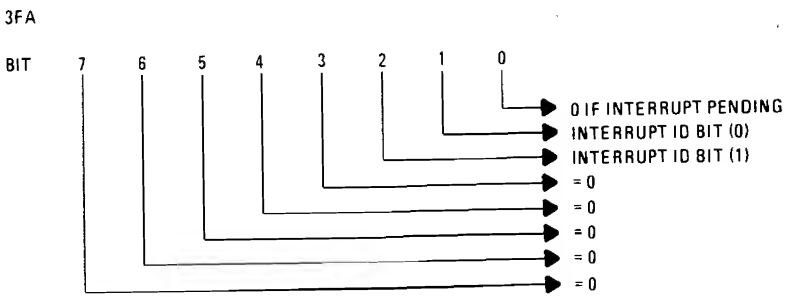
Bit 6: This bit is the Transmitter Shift Register Empty (TSRE) indicator. Bit 6 is set to a logic 1 whenever the Transmitter Shift Register is idle. It is reset to logic 0 upon a data transfer from the Transmitter Holding Register to the Transmitter Shift Register. Bit 6 is a read-only bit.

Interrupt Identification Register

The INS8250 has an on-chip interrupt capability that allows for complete flexibility in interfacing to all the popular microprocessors presently available. In order to provide minimum software overhead during data character transfers, the INS8250 prioritizes interrupts into four levels. The four levels of interrupt conditions are as follows: Receiver Line Status (priority 1); Received Data Ready (priority 2); Transmitter Holding Register Empty (priority 3); and MODEM Status (priority 4).

Information indicating that a prioritized interrupt is pending and the type of that interrupt are stored in the Interrupt Identification Register (refer to Table 5). The Interrupt Identification Register (IIR), when addressed during chip-select time, freezes the highest priority interrupt pending and no other interrupts are acknowledged until that particular interrupt is serviced by the CPU. The contents of the IIR are indicated and described below.

Interrupt Identification Register (IIR)



Bit 0: This bit can be used in either a hardwired prioritized or polled environment to indicate whether an interrupt is pending. When bit 0 is a logic 0, an interrupt is pending and the IIR contents may be used as a pointer to the appropriate interrupt service routine. When bit 0 is a logic 1, no interrupt is pending and polling (if used) continued.

Bits 1 and 2: These two bits of the IIR are used to identify the highest priority interrupt pending as indicated in Table 5.

Bits 3 through 7: These five bits of the IIR are always logic 0.

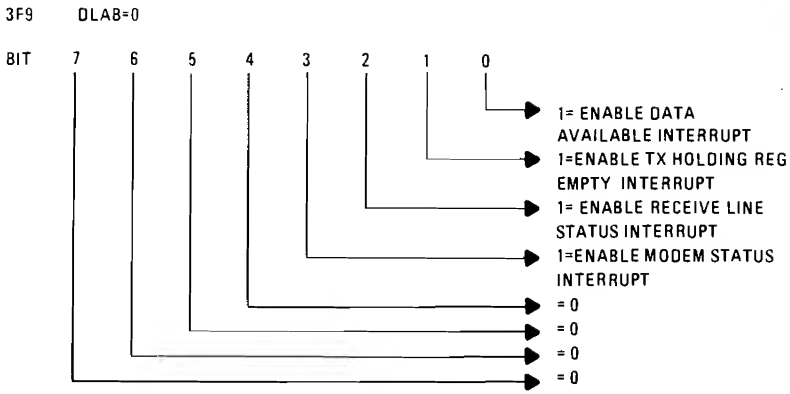
Table 24. Interrupt Control Functions (Asynchronous)

Interrupt ID Register			Interrupt Set and Reset Functions			
Bit 2	Bit 1	Bit 0	Priority Level	Interrupt Type	Interrupt Source	Interrupt Reset Control
0	0	1	–	None	None	–
1	1	0	Highest	Receiver Line Status	Overrun Error or Parity Error or Framing Error or Break Interrupt	Reading the Line Status Register
1	0	0	Second	Received Data Available	Receiver Data Available	Reading the Receiver Buffer Register
0	1	0	Third	Transmitter Holding Register Empty	Transmitter Holding Register Empty	Reading the IIR Register (if source of interrupt) or Writing into the Transmitter Holding Register
0	0	0	Fourth	MODEM Status	Clear to Send or Data Set Ready or Ring Indicator or Received Line Signal Detect	Reading the MODEM Status Register

Interrupt Enable Register

This 8-bit register enables the four types of interrupt of the INS8250 to separately activate the chip Interrupt (INTRPT) output signal. It is possible to totally disable the interrupt system by resetting bits 0 through 3 of the Interrupt Enable Register. Similarly, by setting the appropriate bits of this register to a logic 1, selected interrupts can be enabled. Disabling the interrupt system inhibits the Interrupt Identification Register and the active (high) INTRPT output from the chip. All other system functions operate in their normal manner, including the setting of the Line Status and MODEM Status Registers. The contents of the Interrupt Enable Register are indicated and described below.

Interrupt Enable Register (IER)



Bit 0: This bit enables the Received Data Available Interrupt when set to logic 1.

Bit 1: This bit enables the Transmitter Holding Register Empty Interrupt when set to logic 1.

Bit 2: This bit enables the Receiver Line Status Interrupt when set to logic 1.

Bit 3: This bit enables the MODEM Status Interrupt when set to logic 1.

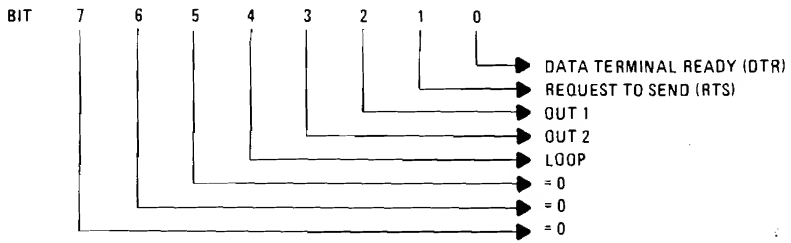
Bits 4 through 7: These four bits are always logic 0.

MODEM Control Register

This 8-bit register controls the interface with the MODEM or data set (or a peripheral device emulating a MODEM). The contents of the MODEM Control Register are indicated and described below.

MODEM Control Register (MCR)

3FC



Bit 0: This bit controls the Data Terminal Ready (DTR) output. When bit 0 is set to a logic 1, the DTR output is forced to a logic 0. When bit 0 is reset to a logic 0, the DTR output is forced to a logic 1.

Note: The DTR output of the INS8250 may be applied to an EIA inverting line driver (such as the DS1488) to obtain the proper polarity input at the succeeding MODEM or data set.

Bit 1: This bit controls the Request to Send (RTS) output. Bit 1 affects the RTS output in a manner identical to that described above for bit 0.

Bit 2: This bit controls the Output 1 (OUT 1) signal, which is an auxiliary user-designated output. Bit 2 affects the OUT 1 output in a manner identical to that described above for bit 0.

Bit 3: This bit controls the Output 2 (OUT 2) signal, which is an auxiliary user-designated output. Bit 3 affects the OUT 2 output in a manner identical to that described above for bit 0.

Bit 4: This bit provides a loopback feature for diagnostic testing of the INS8250. When bit 4 is set to logic 1, the following occur: the transmitter Serial Output (SOUT) is set to the Marking (logic 1) state; the receiver Serial Input (SIN) is disconnected; the output of the Transmitter Shift Register is "looped back" into the Receiver Shift Register input; the four MODEM Control inputs (CTS,DSR, RLSD, and RI) are disconnected; and the four MODEM Control outputs (DTR,RTS,OUT 1, and OUT 2) are internally connected to the four MODEM Control inputs. In the diagnostic mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit- and receive-data paths of the INS8250.

In the diagnostic mode, the receiver and transmitter interrupts are fully operational. The MODEM Control Interrupts are also operational but the interrupts' sources are now the lower four bits of the MODEM Control Register instead of the four MODEM Control inputs. The interrupts are still controlled by the Interrupt Enable Register.

The INS8250 interrupt system can be tested by writing into the lower four bits of the MODEM Status Register. Setting any of these bits to a logic 1 generates the appropriate interrupt (if enabled). The resetting of these interrupts is the same as in normal INS8250 operation. To return to normal operation, the registers must be reprogrammed for normal operation and then bit 4 of the MODEM Control Register must be reset to logic 0.

Bits 5 through 7: These bits are permanently set to logic 0.

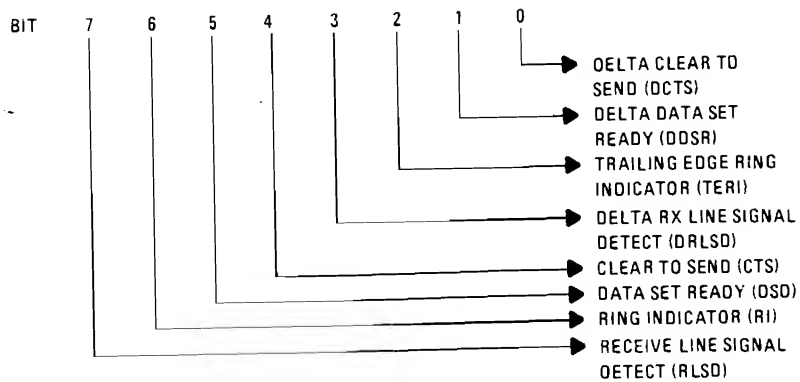
MODEM Status Register

This 8-bit register provides the current state of the control lines from the MODEM (or peripheral device) to the CPU. In addition to this current-state information, four bits of the MODEM Status Register provide change information. These bits are set to a logic 1 whenever a control input from the MODEM changes state. They are reset to logic 0 whenever the CPU reads the MODEM Status Register.

The content of the MODEM Status Register are indicated and described below.

MODEM Status Register (MSR)

3FE



Bit 0: This bit is the Delta Clear to Send (DCTS) indicator. Bit 0 indicates that the CTS input to the chip has changed state since the last time it was read by the CPU.

Bit 1: This bit is the Delta Data Set Ready (DDSR) indicator. Bit 1 indicates that the DSR input to the chip has changed state since the last time it was read by the CPU.

Bit 2: This bit is the Trailing Edge of Ring Indicator (TERI) detector. Bit 2 indicates that the RI input to the chip has changed from an On (logic 1) to an Off (logic 0) condition.

Bit 3: This bit is the Delta Received Line Signal Detector (DRLSD) indicator. Bit 3 indicates that the RLSD input to the chip has changed state.

Note: Whenever bit 0, 1, 2, or 3 is set to a logic 1, a MODEM Status interrupt is generated.

Bit 4: This bit is the complement of the Clear to Send (CTS) input. If bit 4 (loop) of the MCR is set to a 1, this bit is equivalent to RTS in the MCR.

Bit 5: This bit is the complement of the Data Set Ready (DSR) input. If bit 4 of the MCR is set to a 1, this bit is equivalent to DTR in the MCR.

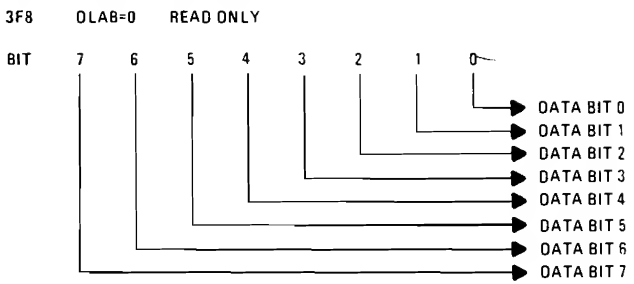
Bit 6: This bit is the complement of the Ring Indicator (RI) input. If bit 4 of the MCR is set to a 1, this bit is equivalent to OUT 1 in the MCR.

Bit 7: This bit is the complement of the Received Line Signal Detect (RLSD) input. If bit 4 of the MCR is set to a 1, this bit is equivalent to OUT 2 of the MCR.

Receiver Buffer Register

The Receiver Buffer Register contains the received character as defined below.

Receiver Buffer Register (RBR)

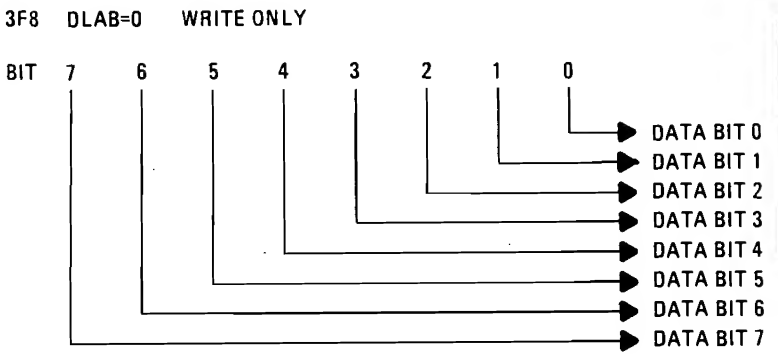


Bit 0 is the least significant bit and is the first bit serially received.

Transmitter Holding Register

The Transmitter Holding Register contains the character to be serially transmitted and is defined below:

Transmitter Holding Register (THR)



Bit 0 is the least significant bit and is the first bit serially transmitted.

Selecting the Interface Format and Adapter Address

The Voltage or Current loop interface and Adapter Address are selected by plugging the programmed shunt modules with the locator dots up or down. See the figure below for the configurations.

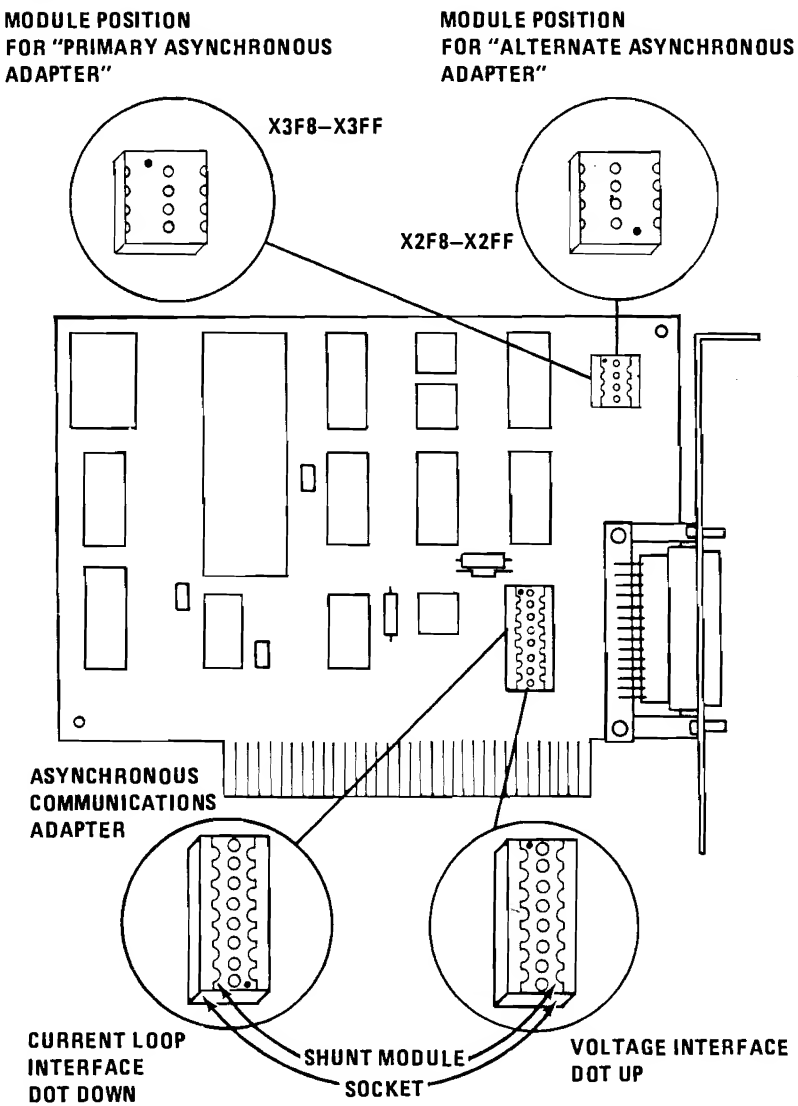
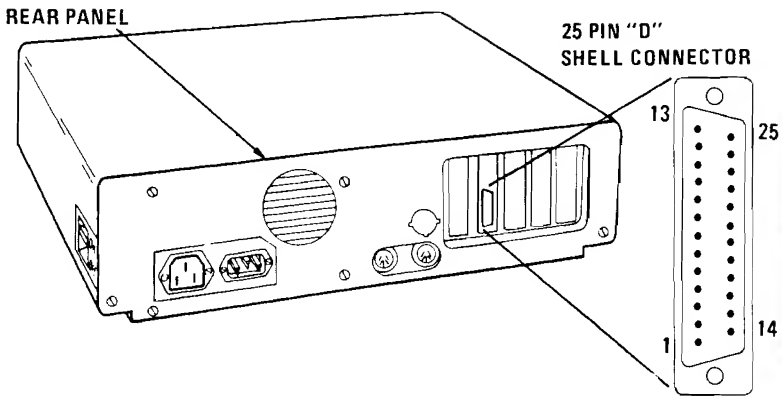


Figure 23. SELECTING THE INTERFACE FORMAT AND ADAPTER ADDRESS

Asynchronous Communications Adapter Connector Interface Specifications



AT STANDARD TTL LEVELS

	Description	Pin	
	NC	1	
←	Transmit Data	2	
→	Receive Data	3	
←	Request to send	4	
→	Clear to send	5	
→	Data set ready	6	
	Signal ground	7	
→	Carrier detect	8	
←	+Transmit current loop return (20 ma)	9	
	NC	10	
←	−Transmit current loop data (20 ma)	11	
	NC	12	Asynchronous Communications Adapter (RS232C)
External Device	NC	13	
	NC	14	
	NC	15	
	NC	16	
	NC	17	
→	+Receive current loop data (20 ma)	18	
	NC	19	
←	Data Terminal Ready	20	
	NC	21	
→	Ring Indicate	22	
	NC	23	
	NC	24	
→	−Receive current loop return (20 ma)	25	

NOTE: To avoid inducing voltage surges on interchange circuits, signals from interchange circuits shall not be used to drive inductive devices, such as relay coils.

NOTES

Prototype Card

The Prototype Card is 4.2 inches high x 13.2 inches long and plugs into a System Expansion Slot. All system control signals and voltage requirements are provided through a 2 x 31 position card edge tab.

The card contains a voltage bus (+5V) and a ground bus (0V). Each bus borders the card, with the voltage bus on the back (pin side) and the ground bus on the front (component side). A system interface design is also provided on the Prototype Card. The logic diagram for this interface is provided in Appendix D. The Prototype Card can also accommodate a D shell connector if it is needed. The connector size can range from a 9 to a 37 position connector.

Note: Install all components on the component side of the Prototype Card. The total width of the card including components should not exceed 0.700". If these specifications are not met, components on the Prototype Card may touch other cards plugged into adjacent slots.

Prototype Card Block Diagram

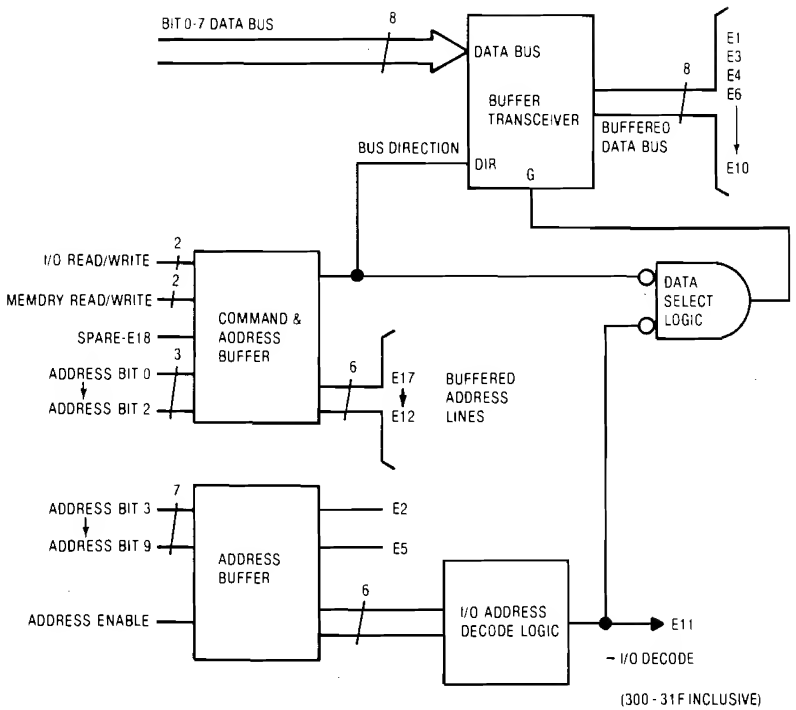


Figure 25. PROTOTYPE CARD BLOCK DIAGRAM

I/O Channel Interface

The Prototype Card has two layers screened onto it (one on the front and one on the back). It also has 3,909 plated through holes that are .040" in size and have a .060" pad which are located on a 0.10" grid. There are 37 plated through holes that are .048" in size. These holes are located at the rear of the card (viewed as if installed in the machine). These 37 holes are used for a 9 to 37 position D shell connector. The card also has 5 holes that are 0.125" in size. One hole is located just above the two rows of D shell connector holes, and the other four are located in the corners of the board (one in each corner).

Prototype Card Layout

The component side has the ground bus (.050" wide) screened on it and card edge tabs that are labeled A1 through A31.

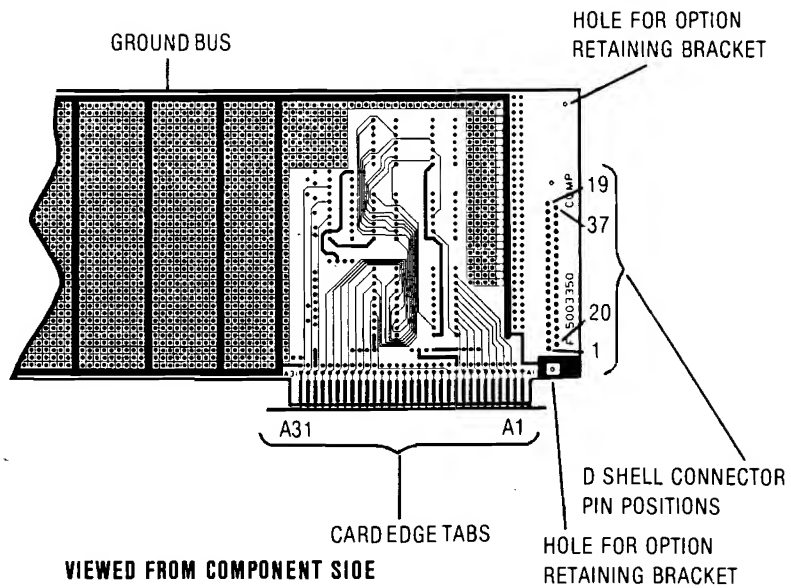
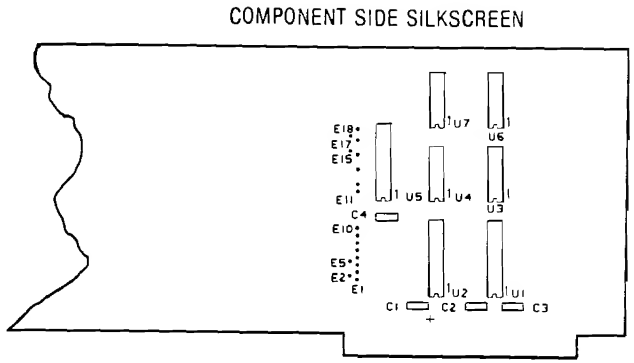


Figure 26. I/O CHANNEL INTERFACE

The component side also has a silk screen printed on it that is used as a component guide for the I/O interface.



VIEWED FROM COMPONENT SIDE

The pin side has a voltage bus (+5v and .050" wide) screened onto it and card edge tabs that are labeled B1 through B31.

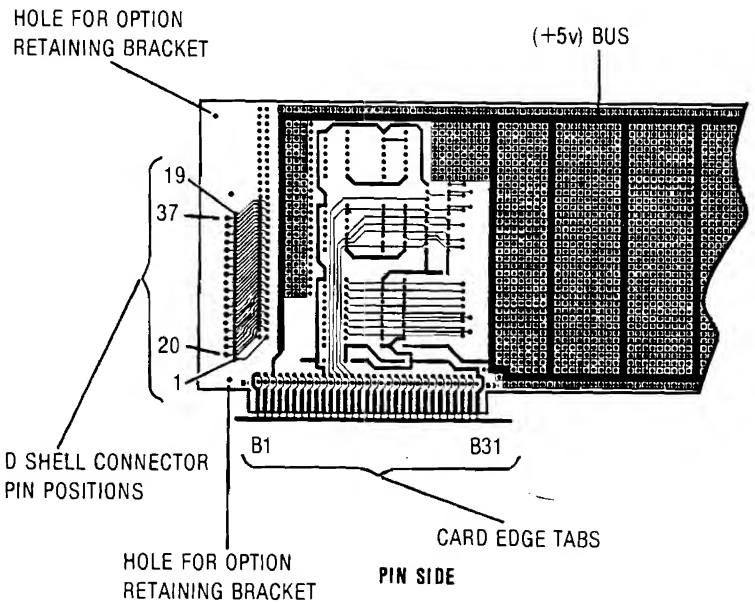


Figure 27. I/O CHANNEL INTERFACE

Each card edge tab is connected to a plated through hole by a .012" land. There are three ground tabs connected to the ground bus by three .012" lands. Also, there are two +5V tabs connected to the voltage bus by two .012" lands.

Prototype Card to System Board Interface

The I/O Channel Description and I/O Channel Diagram are on pages 2-8 through 2-12 of this manual. Also the Prototype Card Interface Logic Diagram is in Appendix D page D-56. If the recommended interface logic is used, the list of TTL type numbers listed below will help you select the necessary components.

Component	TTL Number	Description
U1	74LS245	Octal Bus Transceiver
U2,U5	74LS244	Octal Buffers Line Driver/Line Receivers
U4	74LS04	Hex Inverters
U3	74LS08	Quadruple 2 - Input Positive - And Gate
U6	74LS02	Quadruple 2 - Input Positive - Nor Gate
U7	74LS21	Dual 4 - Input Positive - And Gate
C1		10.0 uf Tantalum Capacitor
C2,C3,C4		.047 uf Ceramic Capacitor

System Loading and Power Limitations

Because of the number of options that may be installed in the Personal Computer, the I/O bus loading should be limited to one Schottky TTL load. If the interface circuitry on the card is used, then this requirement is met.

The power limitations to be observed are located on pages 2-43 through 2-46 of this manual.

Prototype Card External Interface

If a connector is required for the card function, then you should purchase one of the recommended connectors (manufactured by Amp) or equivalent listed below:

Connector Size	Part Number (Amp)
9 pin D shell (Male	205865-1
9 pin D shell (Female)	205866-1
15 pin D shell (Male)	205867-1
15 pin D shell (Female)	205868-1
25 pin D shell (Male)	205857-1
25 pin D shell (Female)	205858-1
37 pin D shell (Male	205859-1
37 pin D shell (Female)	205860-1

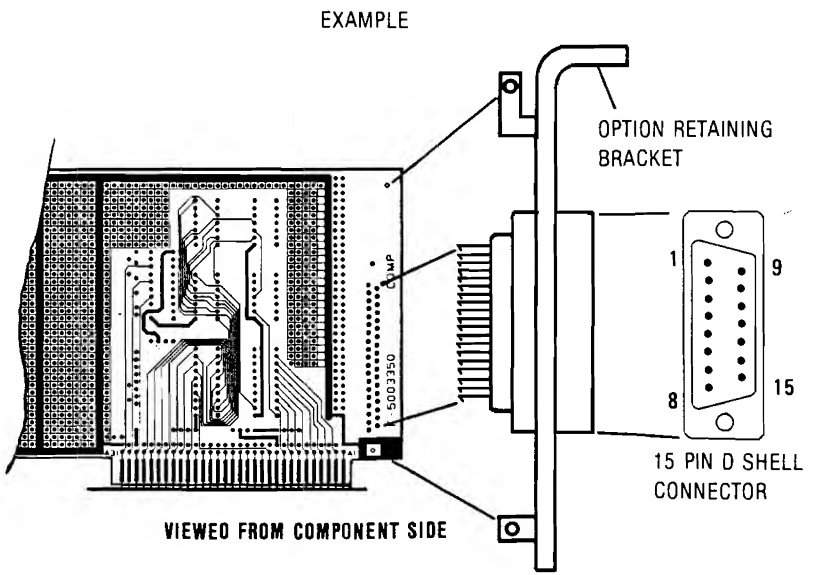


Figure 28. PROTOTYPE CARD EXTERNAL INTERFACE

NOTES

SECTION 3. ROM and SYSTEM USAGE

Contents

ROM BIOS

The ROM resident Basic I/O System (BIOS) provides the device level control of the major I/O devices in the System Unit. The BIOS routines allow the assembly language programmer to perform block (diskette and cassette) or character (Video, communications, keyboard and printer) level I/O operations without any concern for device address and operating characteristics. Additionally, system services such as time of day and memory size determination are provided. The goal is to provide an operational interface to the system and relieve the programmer from concern over hardware device characteristics.

Finally the BIOS interface insulates the user from the hardware allowing new devices to be added to the System Unit, yet retaining the BIOS level interface to the device. In this manner, user programs become transparent to hardware modifications and enhancements. A complete listing of the BIOS is provided in Appendix "A".

Use of BIOS

Access to the BIOS function is through the 8088 software interrupts. Each BIOS entry point is available through its own interrupt, which can be found in the interrupt vector listing. The software interrupts 10H through 1AH each access a different BIOS routine. For example, to determine the amount of memory available in the system,

```
INT     12H
```

will invoke the memory size determination routine in BIOS and return the value to the caller.

Parameter Passing

All parameters passed to and from the BIOS routines go through the 8088 registers. The prologue of each BIOS function indicates the registers used on the call and the return. For the memory size example above, no parameters are passed, and the result, memory size in 1K Byte increments is returned in the AX register.

Where a BIOS function has several possible operations, the AH register is used on input to indicate the desired operation. For example, to set the time of day, the following code is required.


```
MOV     AH, 1               ;function is to set time of
                            day.
MOV     CX,HIGH_COUNT       ;establish the current time.
MOV     DX,LOW_COUNT
INT     1AH                 ;Set the time.
```

While to read the time of day:

```
MOV     AH,0                ;function is to read the time
                            of day.
INT     1AH                 ;read the timer.
```

As a general rule, the BIOS routines preserve all registers except for AX and the flags. Other registers are modified on return only if they are returning a value to the caller. The exact register usage can be seen in the prologue of each BIOS function.

Interrupt Vector Listing

Interrupt Number		Name	BIOS Initialization
0		Divide by Zero	None
1		Single Step	None
2		Non Maskable	NMI_INT (F000:E2C3)
3		Breakpoint	None
4		Dverflow	None
5		Print Screen	PRINT_SCREEN (F000:FF54)
6		Unused	
7		Unused	
8		Time of Day	TIMER_INT (F000:FEA5)
9		Keyboard	KB_INT (F000:E987)
A		Unused	
B	8259	Unused	
C	Interrupt	Unused (Reserved Communications)	
D	Vectors	Unused	
E		Diskette	DISK_INT (F000:EF57)
F		Unused (Reserved Printer)	
10		Video	VIDEO_I O (F000:F065)
11		Equipment Check	EQUIPMENT (F000:F84D)
12		Memory	MEMORY_SIZE_DETERMINE (F000:F841)
13		Diskette	DISKETTE_I O (F000:EC59)
14	BIOS	Communications	RS232_I O (F000:E739)
15	Entry	Cassette	CASSETTE_I O (F000:F859)
16	Points	Keyboard	KEYBOARD_I O (F000:E82E)
17		Printer	PRINTER_I O (F000:EF02)
18		Cassette BASIC	(F600:0000)
19		Bootstrap	BOOT_STRAP (F000:E6F2)
1A		Time of Day	TIME_OF_DAY (F000:FE6E)
1B	User Supplied	Keyboard Break	DUMMY_RETURN (F000:FF53)
1C	Routines	Timer Tick	DUMMY_RETURN (F000:FF53)
1D		Video Initialization	VIDEO_PARMS (F000:F0A4)
1E	BIOS	Diskette Parameters	DISK_BASE (F000:EFC7)
1F	Parameters	Video Graphics Chars	None

NOTES

Vectors with Special Meanings

Interrupt 1BH – Keyboard Break Address

This vector points to the code to be exercised when the CTRL BREAK keys are depressed on the keyboard. The vector is invoked while responding to the keyboard interrupt, and control should be returned via an IRET instruction. The power on routines initialize this vector to point to an IRET instruction, so that nothing happens when CTRL BREAK keys are depressed unless the application program sets a different value.

Control may be retained by this routine, with the following problems. The BREAK may have occurred during interrupt processing, so that one or more End of Interrupt commands must be set to the 8259 controller. Also, all I/O devices should be reset in case an operation was underway at that time.

Interrupt 1CH – Timer Tick

This vector points to the code to be executed on every tick of the system clock. This vector is invoked while responding to the timer interrupt, and control should be returned via an IRET instruction. The power on routines initialize this vector to point to an IRET instruction, so that nothing happens unless the application modifies the pointer. It is the responsibility of the application to save and restore all registers that will be modified.

Interrupt 1DH – Video Parameters

This vector points to a data region containing the parameters required for the initialization of the 6845 on the video card. Note that there are four separate tables, and all four must be reproduced if all modes of operation are to be supported. The power on routines initialize this vector to point to the parameters contained in the ROM video routine.

Interrupt 1EH – Diskette Parameters

This vector points to a data region containing the parameters required for the diskette drive. The power on routines initialize the vector to point to the parameters contained in the ROM diskette routine. These default parameters represent the specified values for any IBM drives attached to the machine. Changing this parameter block to reflect the specifications of the other drives attached may be necessary.

Interrupt 1FH – Graphics Character Extensions

When operating in the graphics modes of the Color/Graphics Monitor Adapter (320 x 200 or 640 x 200), the read/write character interface will form the character from the ASCII code point, using a set of dot patterns. The dot patterns for the first 128 code points are contained in ROM. To access the other 128 code points, this vector must be established to point at a table of up to 1K bytes, where each code point is represented by 8 bytes of graphic information. At power on this vector is initialized to 0:0, and it is the responsibility of the user to change this vector if the additional code points are required.

Other Read/Write Memory Usage

The IBM ROM BIOS routines use 256 bytes of memory starting at absolute 400 to 4FF. Locations 400-407 contain the base addresses of any RS232 cards attached to the system, 0's if none attached. These locations, in order, represent the 0 to 3 values used as the parameter to the RS232 BIOS routine. Locations 408-40F provide the same function, but for the PRINTER.

Memory locations 300-3FF are used as a stack area during the power on initialization, and the bootstrap, when control passed to it from power on. If the user desires the stack in a different area, it must be set by the application.

Note: Use the Interrupt Vector Listing as an aid to locate these topics in the ROM BIOS listing, Appendix "A".

BIOS Programming Tip

When programming with BIOS you should keep in mind that if an error is reported by the diskette code, to reset the diskette adapter and retry the operation. A specified number of retrys should be required on reads to ensure the problem is not due to motor start-up.

BIOS Memory Map

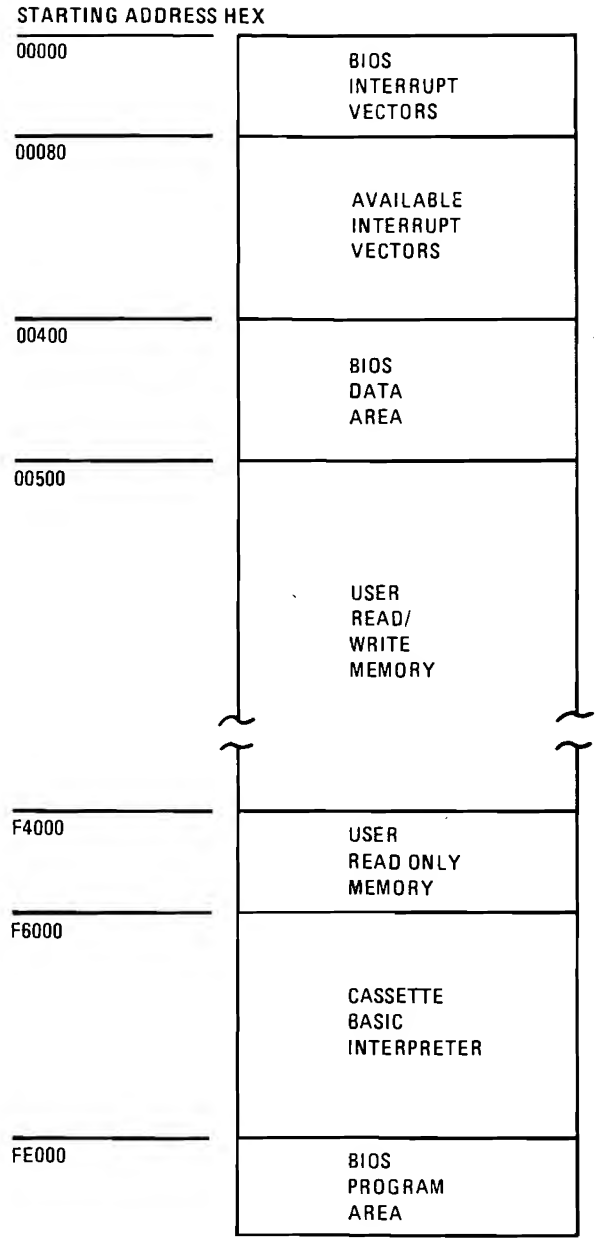


Figure 29. BIOS MEMORY MAP

BIOS Cassette Logic Software Algorithms

Interrupt 15

The cassette routine will be called with the request type in AH and the address of the bytes to be read or written will be specified by (ES):(BX) and the number of bytes to read/write will be specified by (CX). The actual number of bytes read will be returned in (DX). Read block and write block will automatically turn the motor on at the start and off at the end. The requests are as follows:

(AH) = 0	Turn the cassette motor on.
(AH) = 1	Turn the cassette motor off.
(AH) = 2	(Read Block) Read (CX) bytes into memory beginning at address (ES):(BX) and return actual number of bytes read in (DX). Return the cassette status in (AH).
(AH) = 3	(Write Block) Write (CX) bytes onto the cassette beginning at address (DS):(BX). Return the cassette status in (AH).

STATUS:

AH = 00	No errors
AH = 01	CRC-Error (Read Block)
AH = 02	No data transitions
AH = 04	No leader
AH = 80	Invalid command

Note: The carry flag will be set on any error.

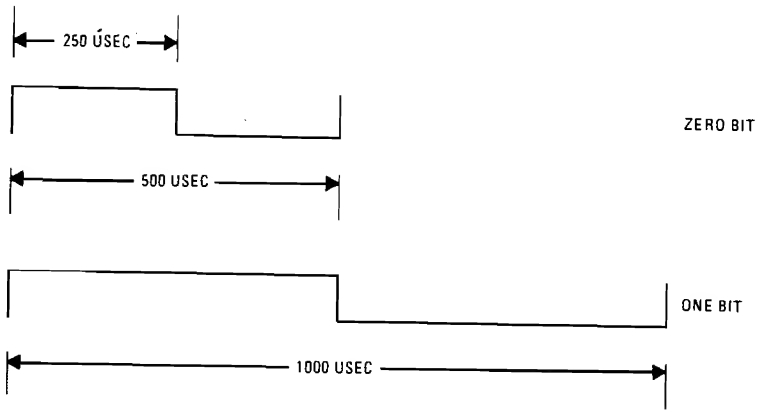
Cassette Write

The WRITE BLOCK routine writes a tape block on the cassette. The tape block is described in Data Record Architecture page (3-10).

The WRITE BLOCK routine turns on the cassette motor and a synchronization bit (0) and then writes 256 bytes of all ones, the leader, to the tape. Next, one or more data blocks are written (depends on number in CX). After each data block of 256 bytes, a two byte CRC is written. The data bytes are taken from the memory location pointed at by ES.

The WRITE BYTE routine disassembles the byte and writes it a bit at a time to the cassette. The method used is to set TIMER 2 to the period of the desired data bit. The timer is set to a period of 1.0 millisecond for a one bit and 0.5 millisecond for a zero bit.

The timer is set mode 3 which means it will output a square wave with period given by its count register. The timer's period is changed on the fly for each data bit to be written to the cassette. If the number of data bytes to be written is not an integral multiple off 256, then after the last desired data byte from memory has been written, the data block will be extended to 256 bytes by writing multiples of the last data byte. The last block will be closed with two CRC bytes as usual. After the last data block, a trailer consisting of four bytes of all one bits will be written. Finally, the motor will be turned off. There are no errors reported by this routine.



Cassette Read

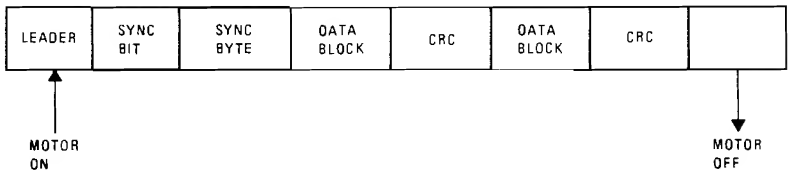
The READ BLOCK routine turns on the cassette motor and then delays for approximately 0.5 secs for it to come up to speed.

The READ BLOCK routine then searches for leader and must detect all one bits for approximately 1/4 of leader length before it can look for the sync byte. If a correct sync byte (X'16') is not found, the routine goes back and searches for leader again. The data is read a bit at a time and assembled into bytes. After each byte is assembled it is written into memory at location ES:BX and then BX is incremented by one.

After each multiple of 256 data bytes are read, the CRC is read and compared to the CRC generated. If a CRC error is detected, the routine will exit with the carry flag set to indicate an error and status (AH)=01 for CRC error. DX will contain the number of bytes written into memory.

Note: The Time of Day Interrupt (IRQ0) is disabled during the cassette read operation.

Data Record Architecture



1. Leader 256 bytes (of ones)
2. Sync byte ASCII Sync Char (X'16')
3. Sync byte (X'16')
4. Data Blocks 256 bytes
5. CRC – 2 bytes – for each data block

Error Recovery

Error recovery is handled by software. A cyclic redundancy check (CRC) is used to detect errors. The polynomial used is:

$G(X) - X^{16} \ll X^{12} \ll X^5$ X 1

Which is the polynomial used by the SDLC interface. Essentially, as bits are written/read from tape, they are passed through the CRC-register in software. After a block of data is written, the complemented value of the calculated CRC-register is written on tape. On reading the cassette data, the CRC bytes are read and compared to the generated CRC value. If the read CRC does not equal the generated one, the processor's carry flag is set and status (AH) is set to X'01' to indicate a CRC error has occurred. Also, the routine is exited on CRC error.

Keyboard Encoding and Usage

Encoding

The keyboard routine provided by IBM in ROM BIOS is responsible for converting the keyboard scan codes into what will be termed "Extended ASCII".

Extended ASCII encompasses one byte character codes with possible values of 0-255, an extended code for certain extended keyboard functions and functions that are handled within the keyboard routine or through interrupts.

Character Codes

The following character codes are passed through the BIOS keyboard routine to the system or application program. A "-1" means the combination is suppressed in the keyboard routine. The codes are returned in AL. See Appendix C for exact codes. Use keyboard Scan Code diagram for reference page 2-23.

Table 25. Character Codes (U.S. Keyboard Layout)

KEY #	BASE CASE	UPPER CASE	CTRL	ALT
1	ESC	ESC	ESC	-1
2	1	½	-1	Note 1
3	2	@	NUL (000) Note 1	Note 1
4	3	#	-1	Note 1
5	4	$	-1	Note 1
6	5	%	-1	Note 1
7	6	^	RS (030)	Note 1
8	7	&	-1	Note 1
9	8	*	-1	Note 1
10	9	(	-1	Note 1
11	0	)	-1	Note 1
12	-	_	US (031)	Note 1
13	=	+	-1	Note 1
14	Backspace (008)	Backspace (008)	DEL(127)	-1
15	→| (009)	|← (Note 1)	-1	-1
16	q	Q	DC1 (017)	Note 1
17	w	W	ETB (023)	Note 1
18	e	E	ENQ (005)	Note 1
19	r	R	DC2 (018)	Note 1
20	t	T	DC4 (020)	Note 1
21	y	Y	EM (025)	Note 1
22	u	U	NAK (021)	Note 1
23	i	I	HT (009)	Note 1
24	o	O	SI (015)	Note 1
25	p	P	DLE (016)	Note 1
26	[	{	ESC (027)	-1
27	]	}	GS (029)	-1

Table 25. Character Codes (continued)

KEY #	BASE CASE	UPPER CASE	CTRL	ALT
28	CR	CR	LF (010)	–1
29CTRL	–1	–1	–1	–1
30	a	A	SOH (001)	Note 1
31	s	S	DC3 (019)	Note 1
32	d	D	EOT (004)	Note 1
33	f	F	ACK (006)	Note 1
34	g	G	BEL (007)	Note 1
35	h	H	BS (008)	Note 1
36	j	J	LF (010)	Note 1
37	k	K	VT (011)	Note 1
38	l	L	FF (012)	Note 1
39	;	:	–1	–1
40	'	"	–1	–1
41	`	~	–1	–1
42SHIFT	–1	–1	–1	–1
43	\	|	FS (028)	–1
44	z	Z	SUB (026)	Note 1
45	x	X	CAN (024)	Note 1
46	c	C	ETX (003)	Note 1
47	v	V	SYN (022)	Note 1
48	b	B	STX (002)	Note 1
49	n	N	SO (014)	Note 1
50	m	M	CR (013)	Note 1
51	,	<	–1	–1
52	.	>	–1	–1
53	/	?	–1	–1
54SHIFT	–1	–1	–1	–1
55	*	(Note 2)	(Note 1)	–1
56ALT	–1	–1	–1	–1
57	SP	SP	SP	SP
58CAPS LOCK	–1	–1	–1	–1
59	NUL (Note 1)	NUL (Note 1)	NUL (Note 1)	NUL (Note 1)
60	NUL (Note 1)	NUL (Note 1)	NUL (Note 1)	NUL (Note 1)
61	NUL (Note 1)	NUL (Note 1)	NUL (Note 1)	NUL (Note 1)
62	NUL (Note 1)	NUL (Note 1)	NUL (Note 1)	NUL (Note 1)
63	NUL (Note 1)	NUL (Note 1)	NUL (Note 1)	NUL (Note 1)
64	NUL (Note 1)	NUL (Note 1)	NUL (Note 1)	NUL (Note 1)
65	NUL (Note 1)	NUL (Note 1)	NUL (Note 1)	NUL (Note 1)
66	NUL (Note 1)	NUL (Note 1)	NUL (Note 1)	NUL (Note 1)
67	NUL (Note 1)	NUL (Note 1)	NUL (Note 1)	NUL (Note 1)
68	NUL (Note 1)	NUL (Note 1)	NUL (Note 1)	NUL (Note 1)
69NUM LOCK	–1	–1	Pause (Note 2)	–1
70SCROLL LOCK	–1	–1	Break (Note 2)	–1

Note 1: Refer to Extended Codes Page (3-13).

Note 2: Refer to Special Handling Page (3-15).

Keys 71-83 have meaning only in base case, in NUMLOCK (or shifted) states, or in CTRL state. It should be noted that the shift key temporarily reverses the current NUMLOCK state.

KEY #	NUM LOCK	BASE CASE	ALT	CTRL
71	7	Home (Note 1)	Note 1	Clear Screen
72	8	↑ (Note 1)	Note 1	–1
73	9	PageUp (Note 1)	Note 1	Top of Text & Home
74	–	–	–1	–1
75	4	← (Note 1)	Note 1	Reverse Word (Note 1)
76	5	–1	Note 1	–1
77	6	→ (Note 1)	Note 1	Adv Word (Note 1)
78	+	+	–1	–1
79	1	End (Note 1)	Note 1	Erase to EOL (Note 1)
80	2	↓ (Note 1)	Note 1	–1
81	3	PageDown (Note 1)	Note 1	Erase to EOS (Note 1)
82	0	INS	Note 1	–1
83		DEL (Notes 1,2)	Note 2	Note 2

Note 1: Refer to Extended Codes Page (3-13).
Note 2: Refer to Special Handling Page (3-15).

Extended Codes

A. Extended Functions

For certain functions that cannot be represented in the standard ASCII code, an extended code is used. A character code of 000 (NUL) is returned in AL. This indicates that the system or application program should examine a second code that will indicate the actual function. Usually, but not always, this second code is the scan code of the primary key that was pressed. This code is returned in AH.

ROM

Table 26. Keyboard Extended Functions

SECOND CODE	FUNCTION
3	NUL Character
15	←
16-25	ALT Q, W, E, R, T, Y, U, I, O, P
30-38	ALT A, S, D, F, G, H, J, K, L
44-50	ALT Z, X, C, V, B, N, M
59-68	F1-F10 Function Keys Base Case
71	Home
72	↑
73	Page Up & Home Cursor
75	←
77	→
79	End
80	↓
81	Page Down & Home Cursor
82	INS
83	DEL
84-93	F11-F20 (Upper Case F1-F10)
94-103	F21-F30 (CTRL F1-F10)
104-113	F31-F40 (ALT F1-F10)
114	CTRL PRTSC (Start/Stop Echo to Printer) Key 55
115	CTRL ← Reverse Word
116	CTRL → Advance Word
117	CTRL END Erase EOL
118	CTRL PG DN Erase EOS
119	CTRL HOME Clear Screen and home
120-131	ALT 1, 2, 3, 4, 5. 6, 7, 8, 9, 0, -, = (Keys 2-13)
132	CTRL PG UP TOP 25 Lines of Text & Home Cursor

B. Shift States

Most shift states are handled within the keyboard routine transparent to the system or application program. In any case, the current set of active shift states are available by calling an entry point in the ROM keyboard routine. The following keys result in altered shift states:

Shift – Temporarily shifts keys 2-13, 15-27, 30-41, 43-53, 55, 59-68 to upper case (lower case if in CAPSLOCK state). Temporarily reverses NUMLOCK/NONUMLOCK state of keys 71-73, 75, 77, 79-83.

CTRL – Temporarily shifts keys 3, 7, 12, 14, 16-28, 30-38, 43-50, 55, 59-71, 73, 75, 77, 79, 81 to CTRL state. Used with ALT and DEL to cause "system reset" function described in Section I.3. Used with SCROLL LOCK to cause "break" function described in Section I.3. Used with NUMLOCK to cause "pause" function described in Section I.3.

ALT – Temporarily shifts keys 2-13, 16-25, 30-38, 44-50, and 59-68 to ALT state. Used with CTRL and DEL to cause system reset function described in Section I.3.

ALT has a special use to allow the user to enter any character code (0-255) into the system from the keyboard. The user holds down the ALT key and types the decimal value of characters using the numeric keyboard (keys 71-73, 75-77, 79-82). The ALT key is then released. If more than three digits are typed, a modulo 256 result is created. These three keys are interpreted as a character code (000-255) and are transmitted through the keyboard routine to the system or application program. ALT is handled internal to keyboard routine.

CAPS LOCK – Shifts keys 16-25, 30-38, 44-50 to upper case. A second depression of CAPS LOCK reverses the action. Handled internal to keyboard routine.

NUM LOCK – Shifts keys 71-73, 75-77, 79-83 to numeric state. A second depression of NUM LOCK reverses the action. Handled internal to keyboard routine.

SCROLL LOCK – Interpreted by appropriate application programs as indicating that the use of the cursor control keys should cause windowing over the text rather than cursor movement. A second depression of SCROLL LOCK reverses the action. The keyboard routine simply records the current shift state of SCROLL LOCK. It is up to the system or application program to perform the function.

C. Shift Key Priorities and Combinations

If combinations of ALT, CTRL and SHIFT are pressed and only one is valid, the precedence is as follows: Highest is ALT, then CTRL, then SHIFT. The only valid combination is ALT CTRL, which is used in system reset.

Special Handling

A. System Reset

The combination of ALT CTRL DEL (Key 83) will result in the keyboard routine initiating the equivalent of a system reset/reboot. Handled internal to keyboard routine.

B. **Break**

The combination CTRL BREAK will result in the keyboard routine signaling interrupt -1A. Also, the extended characters (AL = 00H, AH = 00H) will be returned.

Power up initialization, this interrupt is set up to cause the break sequence to be ignored. It is up to the system or application initialization code to change the interrupt vector in order to support an actual "break" function.

C. **Pause**

The combination CTRL NUM-LOCK will cause the keyboard interrupt routine to loop, waiting for any key except NUM-LOCK to be pressed. This provides a system/application transparent method of suspending list/print/etc. temporarily, and then resuming. The "Unpause" key is thrown away. Handled internal to keyboard routine.

D. The following keys will have their typematic action suppressed by the keyboard routine: CTRL, SHIFT, ALT, NUM-LOCK, SCROLL-LOCK, CAPS LOCK, INS.

E. **Print Screen**

The combination SHIFT-PRINT SCREEN (Key 55) will result in an interrupt invoking the print screen routine. This routine works in alpha/graphics mode, with unrecognizable characters printing as blanks.

The keyboard routine does its own buffering. The buffer is big enough to support a fast typist. If a key is entered when the buffer is full, the key will be ignored and the "bell" will be sounded.

Keyboard Usage

This section is intended to outline a set of guidelines for key usage when performing commonly used functions.

Table 27. Keyboard - Commonly Used Functions

FUNCTION	KEY(S)	COMMENT
Home Cursor	HOME	Editors; word processors
Return to outermost menu	HOME	Menu driven applications
Move cursor up	↑	Full screen editor, word processor
Page up, scroll backwards 25 lines & home	PG UP	Editors; word processors
Move cursor left	← Key 75	Text, command entry
Move cursor right	→	Text, command entry
Scroll to end of text Place cursor at end of line	END	Editors; word processors
Move cursor down	↓	Full screen editor, word processor
Page down, scroll forwards 25 lines & home	PG DN	Editors; word processors
Start/Stop insert text at cursor, shift text right in buffer	INS	Text, command entry
Delete character at cursor	DEL	Text, command entry
Destructive backspace	← Key 14	Text, command entry
Tab forward	→|	Text entry
Tab reverse	|←	Text entry
Clear screen and home	CTRL HOME	Command entry
Scroll up	↑	In scroll lock mode
Scroll down	↓	In scroll lock mode
Scroll left	←	In scroll lock mode
Scroll right	→	In scroll lock mode
Delete from cursor to EOL	CTRL END	Text, command entry
Exit/Escape	ESC	Editor, 1 level of menu, etc
Start/Stop Echo screen to printer	PRTSC CTRL K55	Any time
Delete from cursor to EOS	CTRL PG DN	Text, command entry

Table 27. Keyboard – Commonly Used Functions (continued)

FUNCTION	KEY(S)	COMMENT
Advance word	CTRL →	Text entry
Reverse word	CTRL ←	Text entry
Window Right	CTRL →	When text is too wide to fit screen
Window Left	CTRL ←	When text is too wide to fit screen
Enter insert mode	INS	Line editor
Exit insert mode	INS	Line editor
Cancel current line	ESC	Command entry, text entry
Suspend system (pause)	CTRL NUMLOCK	Stop list, stop program, etc. Resumes on any key
Break interrupt	CTRL BREAK	Interrupt current process
System reset	ALT CTRL DEL	Reboot
Top of document and home cursor	CTRL PG UP	Editors, word processors
Standard Function Keys	F1–F10	Primary function keys
Secondary function keys	SHIFT F1–F10 CTRL F1–F10 ALT F1–F10	Extra function keys if 10 are not sufficient
Extra function keys	ALT Keys 2–13 (1–9,0,–,=)	Used when stickers are put along top of keyboard
Extra function keys	ALT A–Z	Used when function starts with same letter as one of the alpha keys

Table 28. BASIC Screen Editor Special Functions

FUNCTION	KEY
Carriage return	↵
Line feed	CTRL ↵
Bell	CTRL G
Home	HOME
Cursor up	↑
Cursor down	↓
Cursor left	←
Cursor right	→
Advance one word	CTRL →
Reverse one word	CTRL ←
Insert	INS
Delete	DEL
Clear screen	CTRL HOME
Freeze output	CTRL NUMLOCK
Tab advance	→|
Stop execution (break)	CTRL BREAK
Delete current line	ESC
Delete to end of line	CTRL END
Position cursor to end of line	END

Table 29. DOS Special Functions

FUNCTION	KEY
Suspend	CTRL NUMLOCK
Echo to printer	CTRL-PRTSC (Key 55 any case)
Stop echo to printer	CTRL-PRTSC (Key 55 any case)
Exit current function (break)	CTRL BREAK
Backspace	← Key 14
Line feed	CTRL ↵
Cancel line	ESC
Copy character	F1 or →
Copy till match	F2
Copy remaining	F3
Skip character	DEL
Skip until match	F4
Enter insert mode	INS
Exit insert mode	INS
Make new line the template	F5
String separator in REPLACE	F6
End of file in keyboard input	F6

NOTES

Low Memory Maps (0-'0600'x)

Table 30. Interrupt Vectors (0-7F)

ADDRESS HEX	INTERRUPT HEX	FUNCTION
0–3	0	Divide by Zero
4–7	1	Single step
8–B	2	Non-Maskable Interrupt (NMI)
C–F	3	Break Point Instruction ('CC'x)
10–13	4	Overflow
14–17	5	Print Screen
18–1F	6,7	Reserved
20–23	8	Timer (18.2 per second)
24–27	9	Keyboard Interrupt
28–37	A,B,C,D	Reserved
38–3B	E	Diskette Interrupt
3C–3F	F	Reserved
40–43	10	Video I/O Call
44–47	11	Equipment Check Call
48–4B	12	Memory Check Call
4C–4F	13	Diskette I/O Call
50–53	14	RS232 I/O Call
54–57	15	Cassette I/O Call
58–5B	16	Keyboard I/O Call
5C–5F	17	Printer I/O Call
60–63	18	ROM Basic Entry Code
64–67	19	Boot Strap Loader
68–6B	1A	Time of Day Call
6C–6F	1B	Get Control on Keyboard Break: Note 1
70–73	1C	Get Control on timer interrupt: Note 1
74–77	1D	Pointer to video initialization table: Note 2
78–7B	1E	Pointer to diskette parameter table: Note 2
7C–7F	1F	Pointer to table (1KB) for graphics character Generator for ASCII 128–255. Defaults to 0:0

Notes: (1) Initialized at power up to point to an IRET Instruction.
(2) Initialized at power up to point to tables in ROM.

Table 31. BASIC & DOS Reserved Interrupts (80-3FF)

ADDRESS HEX	INTERRUPT HEX	FUNCTION
80–83	20	DOS Program Terminate
84–87	21	DOS Function Call
88–8B	22	DOS Terminate Address
8C–8F	23	DOS CTRL–BRK Exit Address
90–93	24	DOS Fatal Error Vector
94–97	25	DOS Absolute Disk read
98–9B	26	DOS Absolute Disk write
9C–9F	27	DOS Terminate, Fix in Storage
A0–FF	28–3F	Reserved for DOS
100–1FF	40–7F	Not Used
200–217	80–85	Reserved By BASIC
218–3C3	86–F0	Used by BASIC Interpreter while BASIC is Running.
3C4–3FF	F1–FF	Not Used

Table 32. Reserved Memory Locations (400-5FF)

ADDRESS HEX	MODE	FUNCTION
400–48F	ROM BIOS	See BIOS Listing
490–4CF	DOS	Used by DOS Mode Command
4D0–4EF		Reserved
4F0–4FF		Reserved as Intra-Application Communication area for any application.
500–5FF		Reserved for DOS and BASIC
500	DOS	Print Screen status flag store. 0–Print screen not active or successful print screen operation. 1–Print screen in progress. 255–Error encountered during print screen operation.
504	DOS	Single drive mode status byte.
510–511	BASIC	BASIC's segment address store.
512–515	BASIC	Clock interrupt vector segment: offset store.
516–519	BASIC	Break key interrupt vector segment: offset store.
51A–51D	BASIC	Disk error interrupt vector segment: offset store.

BASIC Workspace Variables

If you do DEF SEG (Default workspace segment)

	OFFSET	LENGTH
Line number of current line being executed	X '2E'	2
Line number of last error	X '347'	2
Offset into segment of start of program text	X '30'	2
Offset into of start of variables (end of program text 1–1)	X '358'	2
Keyboard buffer contents if 0–no characters in buffer if 1–characters in buffer if you POKE & H6A, 0 you flush any characters in buffer	X '6A'	1

Example:

```
100 Print   PEEK (&H2E) + 256*PEEK (&H2F)
                    L          H
100         | X '64' | X '00' |
```

NOTES

APPENDICES

Contents

NOTES

APPENDIX A. ROM BIOS LISTINGS

Contents

```
LOC OBJ                 LINE    SOURCE

                          1     $TITLE(ROM BIOS FOR IBM PERSONAL COMPUTER)
                          2     ;--------------------------
                          3     ; EQUATES
                          4     ;--------------------------
 0060                     5     PORT_A          EQU     60H                     ;8255 PORT A ADDR
 0061                     6     PORT_B          EQU     61H                     ;8255 PORT B ADDR
 0062                     7     PORT_C          EQU     62H                     ;8255 PORT C ADDR
 0063                     8     CMD_PORT EQU    63H
 0020                     9     INTA00          EQU     20H                     ;8259 PORT
 0021                    10     INTA01          EQU     21H                     ;8259 PORT
 0020                    11     EOI             EQU     20H
 0040                    12     TIMER           EQU     40H
 0043                    13     TIM_CTL EQU     43H                     ;8253 TIMER CONTROL PORT ADDR
 0040                    14     TIMER0  EQU     40H                     ;8253 TIMER/CNTER 0 PORT ADDR
 0001                    15     TMINT   EQU     01                      ;TIMER 0 INTR RECVD MASK
 0008                    16     DMA08   EQU     08                      ;DMA STATUS REG PORT ADDR
 0000                    17     DMA     EQU     00                      ;DMA CHANNEL 0 ADDRESS REG PORT ADDR
 0540                    18     MAX_PERIOD EQU  540H
 0410                    19     MIN_PERIOD EQU  410H
 0060                    20     KBD_IN  EQU     60H                             ;KEYBOARD DATA IN ADDR PORT
 0002                    21     KBDINT  EQU     02                              ;KEYBOARD INTR MASK
 0060                    22     KB_DATA         EQU     60H     ; KEYBOARD SCAN CODE PORT
 0061                    23     KB_CTL          EQU     61H     ; CONTROL BITS FOR KEYBOARD SENSE DATA
                         24     ;-----------------------------------------
                         25     ; 8088 INTERRUPT LOCATIONS
                         26     ;-----------------------------------------
 0000                    27     ABS0    SEGMENT AT 0
 0000                    28     STG_LOC0        LABEL   BYTE
 0008                    29             ORG     2*4
 0008                    30     NMI_PTR         LABEL    WORD
 0014                    31             ORG     5*4
 0014                    32     INT5_PTR        LABEL   WORD
 0020                    33             ORG     8*4
 0020                    34     INT_ADDR        LABEL   WORD
 0020                    35     INT_PTR         LABEL   DWORD
 0040                    36             ORG     10H*4
 0040                    37     VIDEO_INT       LABEL   WORD
 0074                    38             ORG     1DH*4
 0074                    39     PARM_PTR        LABEL   DWORD           ; POINTER TO VIDEO PARMS
 0078                    40             ORG     01EH*4                  ; INTERRUPT 1EH
 0078                    41     DISK_POINTER    LABEL   DWORD
 007C                    42             ORG     01FH*4                  ; LOCATION OF POINTER
 007C                    43     EXT_PTR LABEL   DWORD           ; POINTER TO EXTENSION
 7C00                    44             ORG     7C00H
 7C00                    45     BOOT_LOCN       LABEL   FAR
                         46     ABS0    ENDS
                         47
                         48     ;----------------------
                         49     ; STACK -- USED DURING INITIALIZATION ONLY
                         50     ;----------------------
                         51     STACK   SEGMENT AT 30H
 0000 (128 ????)         52             DW      128 DUP(?)
 0100                    53     TOS     LABEL   WORD
                         54     STACK   ENDS
                         55
                         56     ;-----------------------------------------
                         57     ; ROM BIOS DATA AREAS
                         58     ;-----------------------------------------
 0040                    59     DATA    SEGMENT AT 40H
 0000 (4 ????)           60     RS232_BASE      DW      4 DUP(?)        ; ADDRESSES OF RS232 ADAPTERS
 0008 (4 ????)           61     PRINTER_BASE    DW      4 DUP(?)        ; ADDRESSES OF PRINTERS
 0010 ????               62     EQUIP_FLAG      DW      ?               ; INSTALLED HARDWARE
 0012 ??                 63     MFG_TST         DB      ?               ; INITIALIZATION FLAG
 0013 ????               64     MEMORY_SIZE     DW      ?               ; MEMORY SIZE IN K BYTES
 0015 ????               65     IO_RAM_SIZE     DW      ?               ; MEMORY IN I/O CHANNEL
                         66     ;-----------------------------------------
                         67     ; KEYBOARD DATA AREAS
                         68     ;-----------------------------------------
 0017 ??                 69     KB_FLAG         DB      ?
                         70
                         71     ;------ SHIFT FLAG EQUATES WITHIN KB_FLAG
                         72
   0080                  73     INS_STATE       EQU     80H     ; INSERT STATE IS ACTIVE
   0040                  74     CAPS_STATE      EQU     40H     ; CAPS LOCK STATE HAS BEEN TOGGLED
   0020                  75     NUM_STATE       EQU     20H     ; NUM LOCK STATE HAS BEEN TOGGLED
   0010                  76     SCROLL_STATE    EQU     10H     ; SCROLL LOCK STATE HAS BEEN TOGGLED
   0008                  77     ALT_SHIFT       EQU     08H     ; ALTERNATE SHIFT KEY DEPRESSED
```



```
LOC OBJ                 LINE    SOURCE

   0004                  78     CTL_SHIFT       EQU     04H     ; CONTROL SHIFT KEY DEPRESSED
   0002                  79     LEFT_SHIFT      EQU     02H     ; LEFT SHIFT KEY DEPRESSED
   0001                  80     RIGHT_SHIFT     EQU     01H     ; RIGHT SHIFT KEY DEPRESSED
                         81
 001B ??                 82     KB_FLAG_1       DB      ?       ; SECOND BYTE OF KEYBOARD STATUS
                         83
   0080                  84     INS_SHIFT       EQU     80H     ; INSERT KEY IS DEPRESSED
   0040                  85     CAPS_SHIFT      EQU     40H     ; CAPS LOCK KEY IS DEPRESSED
   0020                  86     NUM_SHIFT       EQU     20H     ; NUM LOCK KEY IS DEFRESSED
   0010                  87     SCROLL_SHIFT    EQU     10H     ; SCROLL LOCK KEY IS DEPRESSED
   0008                  88     HOLD_STATE      EQU     08H     ; SUSPEND KEY HAS BEEN TOGGLED
                         89
 0019 ??                 90     ALT_INPUT       DB      ?       ; STORAGE FOR ALTERNATE KEYPAD ENTRY
 001A ????               91     BUFFER_HEAD     DW      ?       ; POINTER TO HEAD OF KEYBOARD BUFFER
 001C ????               92     BUFFER_TAIL     DW      ?       ; POINTER TO TAIL OF KEYBOARD BUFFER
 001E (16 ????)          93     KB_BUFFER       DW      16 DUP(?) ; ROOM FOR 15 ENTRIES
 003E                    94     KB_BUFFER_END   LABEL   WORD
                         95
                         96     ;------ HEAD = TAIL INDICATES THAT THE BUFFER IS EMPTY
                         97
   0045                  98     NUM_KEY         EQU     69      ; SCAN CODE FOR NUMBER LOCK
   0046                  99     SCROLL_KEY      EQU     70      ; SCROLL LOCK KEY
   0038                 100     ALT_KEY         EQU     56      ; ALTERNATE SHIFT KEY SCAN CODE
   001D                 101     CTL_KEY         EQU     29      ; SCAN CODE FOR CONTROL KEY
   003A                 102     CAPS_KEY        EQU     58      ; SCAN CODE FOR SHIFT LOCK
   002A                 103     LEFT_KEY        EQU     42      ; SCAN CODE FOR LEFT SHIFT
   0036                 104     RIGHT_KEY       EQU     54      ; SCAN CODE FOR RIGHT SHIFT
   0052                 105     INS_KEY         EQU     82      ; SCAN CODE FOR INSERT KEY
   0053                 106     DEL_KEY         EQU     83      ; SCAN CODE FOR DELETE KET
                        107
                        108     ;-----------------------------------------
                        109     ; DISKETTE DATA AREAS
                        110     ;-----------------------------------------
 003E ??                111     SEEK_STATUS     DB      ?       ; DRIVE RECALIBRATION STATUS
                        112     ;                               BIT 3-0 = DRIVE 3-0 NEEDS RECAL BEFORE
                        113     ;                               NEXT SEEK IF BIT IS = 0
   0080                 114     INT_FLAG        EQU     080H    ; INTERRUPT OCCURRENCE FLAG
 003F ??                115     MOTOR_STATUS    DB      ?       ; MOTOR STATUS
                        116     ;                               BIT 3-0 = DRIVE 3-0 IS CURRENTLY RUNNING
                        117     ;                               BIT 7   = CURRENT OPERATION IS A WRITE, REQUIRES DELAY
 0040 ??                118     MOTOR_COUNT     DB      ?       ; TIME OUT COUNTER FOR DRIVE TURN OFF
   0025                 119     MOTOR_WAIT      EQU     37      ; TWO SECONDS OF COUNTS FOR MOTOR TURN OFF
                        120
                        121     ;
 0041 ??                122     DISKETTE_STATUS DB      ?       ; SINGLE BYTE OF RETURN CODE INFO FOR STATUS
   0080                 123     TIME_OUT        EQU     80H     ; ATTACHMENT FAILED TO RESPOND
   0040                 124     BAD_SEEK        EQU     40H     ; SEEK OPERATION FAILED
   0020                 125     BAD_NEC         EQU     20H     ; NEC CONTROLLER HAS FAILED
   0010                 126     BAD_CRC         EQU     10H     ; BAD CRC ON DISKETTE READ
   0009                 127     DMA_BOUNDARY    EQU     09H     ; ATTEMPT TO DMA ACROSS 64K BOUNDARY
   0008                 128     BAD_DMA         EQU     08H     ; DMA OVERRUN ON OPERATION
   0004                 129     RECORD_NOT_FND  EQU     04H     ; REQUESTED SECTOR NOT FOUND
   0003                 130     WRITE_PROTECT   EQU     03H     ; WRITE ATTEMPTED ON WRITE PROT DISK
   0002                 131     BAD_ADDR_MARK   EQU     02H     ; ADDRESS MARK NOT FOUND
   0001                 132     BAD_CMD         EQU     01H     ; BAD COMMAND PASSED TO DISKETTE I/O
                        133
 0042 (7 ??)            134     NEC_STATUS      DB      7 DUP(?)        ; STATUS BYTES FROM NEC
                        135
                        136     ;-----------------------------------------
                        137     ; VIDEO DISPLAY DATA AREA
                        138     ;-----------------------------------------
 0049 ??                139     CRT_MODE        DB      ?       ; CURRENT CRT MODE
 004A ????              140     CRT_COLS        DW      ?       ; NUMBER OF COLUMNS ON SCREEN
 004C ????              141     CRT_LEN         DW      ?       ; LENGTH OF REGEN IN BYTES
 004E ????              142     CRT_START       DW      ?       ; STARTING ADDRESS IN REGEN BUFFER
 0050 (8 ????)          143     CURSOR_POSN     DW      8 DUP(?)        ; CURSOR FOR EACH OF UP TO 8 PAGES
 0060 ????              144     CURSOR_MODE     DW      ?       ; CURRENT CURSOR MODE SETTING
 0062 ??                145     ACTIVE_PAGE     DB      ?       ; CURRENT PAGE BEING DISPLAYED
 0063 ????              146     ADDR_6845       DW      ?       ; BASE ADDRESS FOR ACTIVE DISPLAY CARD
 0065 ??                147     CRT_MODE_SET    DB      ?       ; CURRENT SETTING OF THE 3X8 REGISTER
 0066 ??                148     CRT_PALLETTE    DB      ?       ; CURRENT PALLETTE SETTING COLOR CARD
                        149
                        150     ;-----------------------------------------
                        151     ; CASSETTE DATA AREA
                        152     ;-----------------------------------------
 0067 ????              153     EDGE_CNT        DW      ?               ;TIME COUNT AT DATA EDGE
 0069 ????              154     CRC_REG         DW      ?               ;CRC REGISTER
```

```
LOC OBJ                 LINE    SOURCE

006B ??                 155     LAST_VAL        DB      ?               ;LAST INPUT VALUE
                        156
                        157     ;--------------------------------------------
                        158     ; TIMER DATA AREA
                        159     ;--------------------------------------------
006C ????               160     TIMER_LOW       DW      ?       ; LOW WORD OF TIMER COUNT
006E ????               161     TIMER_HIGH      DW      ?       ; HIGH WORD OF TIMER COUNT
0070 ??                 162     TIMER_OFL       DB      ?       ; TIMER HAS ROLLED OVER SINCE LAST READ
                        163     ;COUNTS_SEC     EQU     18
                        164     ;COUNTS_MIN     EQU     1092
                        165     ;COUNTS_HOUR    EQU     65543
                        166     ;COUNTS_DAT     EQU     1573040 = 1800B0H
                        167
                        168     ;--------------------------------------------
                        169     ; SYSTEM DATA AREA
                        170     ;--------------------------------------------
0071 ??                 171     BIOS_BREAK      DB      ?       ; BIT 7 = 1 IF BREAK KEY HAS BEEN DEPRESSED
0072 ????               172     RESET_FLAG      DW      ?       ; WORD = 1234H IF KEYBOARD RESET UNDERWAY
                        173     DATA    ENDS
                        174
                        175     ;--------------------------------------------
                        176     ; EXTRA DATA AREA
                        177     ;--------------------------------------------
0050                    178     XXDATA  SEGMENT AT 50H
0000 ??                 179     STATUS_BYTE     DB      ?
                        180     XXDATA  ENDS
                        181
                        182     ;--------------------------------------------
                        183     ; VIDEO DISPLAY BUFFER
                        184     ;--------------------------------------------
B800                    185     VIDEO_RAM       SEGMENT AT 0B800H
0000                    186     REGEN   LABEL   BYTE
0000                    187     REGENW  LABEL   WORD
0000 (16384 ??)         188             DB      16384 DUP(?)
                        189     VIDEO_RAM       ENDS
                        190     ;--------------------------------------------
                        191     ; ROM RESIDENT CODE
                        192     ;--------------------------------------------
F000                    193     CODE    SEGMENT AT 0F000H
0000 (57344 ??)         194             DB      57344 DUP(?)    ; FILL LOWEST 56K
                        195
E000 35373030303531     196             DB      '5700051 COPR. IBM 1981'        ; COPYRIGHT NOTICE
     20434F50522E20
     49424D20313938
     31
                        197
                        198     ;--------------------------------------------
                        199     ; INITIAL RELIABILITY TESTS -- PHASE 1
                        200     ;--------------------------------------------
                        201             ASSUME  CS:CODE,SS:CODE,ES:ABS0,DS:DATA
                        202     ;--------------------------------------------
                        203     ;       DATA DEFINITIONS
                        204     ;--------------------------------------------
E016 0BE0           R   205     C1      DW      C11             ; RETURN ADDRESS
E018 EDE1           R   206     C2      DW      C24             ; RETURN ADDRESS FOR DUMMY STACK
                        207     ;--------------------------------------------
                        208     ;       THIS SUBROUTINE PERFORMS A READ/WRITE STORAGE TEST ON A 16K BLOCK
                        209     ;       OF STORGAGE.
                        210     ;ENTRY REQUIREMENTS:
                        211     ;       ES = ADDRESS OF STORAGE SEGMENT BEING TESTED
                        212     ;       DS = ADDRESS OF STORAGE SEGMENT BEING TESTED
                        213     ;       WHEN ENTERING AT STGTST_CNT, CX MUST BE LOADED WITH THE BYTE COUNT.
                        214     ;EXIT PARAMETERS:
                        215     ;       ZERO FLAG = 0 IF STORAGE ERROR (DATA COMPARE OR PARITY CHECK.  AL=0
                        216     ;                       DENOTES A PARITY CHECK. ELSE AL=XOR'ED BIT PATTERN OF THE
                        217     ;                       EXPECTED DATA PATTERN VS THE ACTUAL DATA READ.
                        218     ;       AX,BX,CX,DX,DI, AND SI ARE ALL DESTROYED.
                        219     ;--------------------------------------------
E01A                    220     STGTST          PROC    NEAR
E01A B90040             221             MOV     CX,4000H                ;SETUP CNT TO TEST A 16K BLK
E01D                    222     STGTST_CNT:
E01D FC                 223             CLD                             ;SET DIR FLAG TO INCREMENT
E01E 8BD9               224             MOV     BX,CX                   ;SAVE BYTE CNT (4K FOR VIDEO OR 16K)
E020 B8FFFF             225             MOV     AX,0FFFFH               ;GET DATA PATTERN TO WRITE
E023 BA55AA             226             MOV     DX,0AA55H               ;SETUP OTHER DATA PATTERNS TO USE
E026 2BFF               227             SUB     DI,DI                   ;DI = OFFSET 0 RELATIVE TO ES REG
E028 F3                 228             REP     STOSB                   ;WRITE STORAGE LOCATIONS
E029 AA
```



```
LOC OBJ                 LINE    SOURCE

E02A                    229     C3:                                     ; STGO1
E02A 4F                 230             DEC     DI                      ;POINT TO LAST BYTE JUST WRITTEN
E02B FD                 231             STD                             ;SET DIR FLAG TO GO BACKWARDS
E02C 8BF7               232     C4:     MOV     SI,DI
E02E 8BCB               233             MOV     CX,BX                   ;SETUP BYTE CNT
E030 AC                 234     C5:     LODSB                           ;READ CHAR FROM STORAGE
E031 32C4               235             XOR     AL,AH                   ;DATA READ AS EXPECTED?
E033 7525               236             JNE     C7                      ;NO - GO TO ERROR ROUTINE
E035 E462               237             IN      AL,PORT_C               ;DIO A PARITY ERROR OCCUR?
E037 24C0               238             AND     AL,0C0H
E039 B000               239             MOV     AL,0                    ;AL=0 DATA COMPARE OK
E03B 7510               240             JNZ     C7
E03D 80FC00             241             CMP     AH,0                    ;READING ZERO PATTERN?
E040 7403               242             JE      C6                      ;CONTINUE READING TILL END
E042 8AC2               243             MOV     AL,DL                   ;GET NEXT DATA PATTERN TO WRITE
E044 AA                 244             STOSB                           ;WRITE IN BYTE LOC WE JUST READ
E045                    245     C6:                                     ; WRITE_NO_MORE
E045 E2E9               246             LOOP    C5                      ;CONTINUE TILL 16K/4K BLOCK TESTED
E047 80FC00             247             CMP     AH,0                    ;ZERO PATTERN WRITTEN TO STG
E04A 740E               248             JE      C7                      ;YES - RETURN TO CALLER
E04C 8AE0               249             MOV     AH,AL                   ;SETUP TO NEW VALUE TO COMPARE
E04E 86F2               250             XCHG    DH,DL                   ;MOVE ZERO DATA PATTERN TO DL
E050 FC                 251             CLD                             ;SET DIR FLAG TO GO FORWARD
E051 47                 252             INC     DI                      ;SET POINTER TO BEG LOCATION
E052 74D8               253             JZ      C4                      ;READ/WRITE FORWARD IN STG
E054 4F                 254             DEC     DI
E055 BA010Q             255             MOV     DX,1                    ;SETUP 01 AND 00 PATTERNS
E058 EBD0               256             JMP     SHORT C3                ;READ/WRITE BACKWARD IN STG
E05A                    257     C7:
E05A C3                 258             RET
                        259     STGTST  ENDP
                        260     ;--------------------------------------------
                        261     ;TEST.01
                        262     ;       8088 PROCESSOR TEST
                        263     ;DESCRIPTION
                        264     ;       VERIFY 8088 FLAGS, REGISTERS AND CONDITIONAL JUMPS
                        265     ;--------------------------------------------
E05B                    266     RESET   LABEL   NEAR
E05B FA                 267     START:  CLI                             ;DISABLE INTERRUPTS
E05C B4D5               268             MOV     AH,0D5H                 ;SET SF, CF, ZF, AND AF FLAGS ON
E05E 9E                 269             SAHF
E05F 734E               270             JNC     ERR01                   ;GO TO ERR ROUTINE IF CF NOT SET
E061 754C               271             JNZ     ERR01                   ;GO TO ERR ROUTINE IF ZF NOT SET
E063 7B4A               272             JNP     ERR01                   ;GO TO ERR ROUTINE IF PF NOT SET
E065 7948               273             JNS     ERR01                   ;GO TO ERR ROUTINE IF SF NOT SET
E067 9F                 274             LAHF                            ;LOAD FLAG IMAGE TO AH
E068 B105               275             MOV     CL,5                    ;LOAD CNT REG WITH SHIFT CNT
E06A D2EC               276             SHR     AH,CL                   ;SHIFT AF INTO CARRY BIT POS
E06C 7341               277             JNC     ERR01                   ;GO TO ERR ROUTINE IF AF NOT SET
E06E B040               278             MOV     AL,40H                  ;SET THE OF FLAG ON
E070 00E0               279             SHL     AL,1                    ;SETUP FOR TESTING
E072 7138               280             JNO     ERR01                   ;GO TO ERR ROUTINE IF OF NOT SET
E074 32E4               281             XOR     AH,AH                   ;SET AH = 0
E076 9E                 282             SAHF                            ;CLEAR SF, CF, ZF, AND PF
E077 7236               283             JC      ERR01                   ;GO TO ERR ROUTINE IF CF ON
E079 7434               284             JZ      ERR01                   ;GO TO ERR ROUTINE IF ZF ON
E07B 7832               285             JS      ERR01                   ;GO TO ERR ROUTINE IF SF ON
E07D 7A30               286             JP      ERR01                   ;GO TO ERR ROUTINE IF PF ON
E07F 9F                 287             LAHF                            ;LOAD FLAG IMAGE TO AH
E080 B105               288             MOV     CL,5                    ;LOAD CNT REG WITH SHIFT CNT
E082 D2EC               289             SHR     AH,CL                   ;SHIFT  AF' INTO CARRY BIT POS
E084 7229               290             JC      ERR01                   ;GO TO ERR ROUTINE IF ON
E086 D0E4               291             SHL     AH,1                    ;CHECK THAT  OF' IS CLEAR
E088 7025               292             JO      ERR01                   ;GO TO ERR ROUTINE IF ON
                        293
                        294               READ/WRITE THE 8088 GENERAL AND SEGMENTATION REGISTERS
                        295               WITH ALL ONE'S AND ZEROES'S.
                        296
E08A B8FFFF             297             MOV     AX,0FFFFH                       ;SETUP ONE'S PATTERN IN AX
E08D F9                 298             STC
E08E 8ED8               299     CB:     MOV     DS,AX                           ;WRITE PATTERN TO ALL REGS
E090 8CD8               300             MOV     AX,DS
E092 8EC3               301             MOV     ES,BX
E094 8CC1               302             MOV     CX,ES
E096 8ED1               303             MOV     SS,CX
E098 8CD2               304             MOV     DX,SS
E09A 8BE2               305             MOV     SP,DX
E09C 8BEC               306             MOV     BP,SP
```

```
LOC OBJ                 LINE    SOURCE

E09E 8BF5               397             MOV     SI,BP
E0A0 8BFE               398             MOV     DI,SI
E0A2 7307               399             JNC     C9                      ; TST1A
E0A4 33C7               400             XOR     AX,DI                   ;PATTERN MAKE IT THRU ALL REGS
E0A6 7507               401             JNZ     ERR01                   ;NO - GO TO ERR ROUTINE
E0A8 F8                 402             CLC
E0A9 73E3               403             JNC     C8
E0AB                    404     C9:                                     ; TST1A
E0AB 08C7               405             OR      AX,DI                   ;ZERO PATTERN MAKE IT THRU?
E0AD 7401               406             JZ      C10                     ;YES - GO TO NEXT TEST
E0AF F4                 407     ERR01:  HLT                             ;HALT SYSTEM
                        408     ;-----------------------------------------
                        409     ;TEST.02
                        410     ;       ROS CHECKSUM TEST I
                        411     ;DESCRIPTION
                        412     ;       A CHECKSUM IS DONE FOR THE 8K ROS MODULE CONTAINING POD AND BIOS.
                        413     ;-----------------------------------------
E0B0                    414     C10:
E0B0 B000               415             MOV     AL,0                    ;DISABLE NMI INTERRUPTS
E0B2 E6A0               416             OUT     0A0H,AL
E0B4 E683               417             OUT     83H,AL                  ;INITIALZE DMA PAGE REG
E0B6 B099               418             MOV     AL,99H                  ;SET 8255 A,C-INPUT,B-OUTPUT
E0B8 E663               419             OUT     CMD_PORT,AL             ;WRITE 8255 CMD/MODE REG
E0BA B0FC               420             MOV     AL,0FCH                 ;DISABLE PARITY CHECKERS AND
E0BC E661               421             OUT     PORT_B,AL               ; GATE SWS SWS,CASS MOTOR OFF
E0BE 2AC0               422             SUB     AL,AL                   ;
E0C0 BAD803             423             MOV     DX,3D8H
E0C3 EE                 424             OUT     DX,AL                   ;DISABLE COLOR VIDEO
E0C4 FEC0               425             INC     AL
E0C6 BAB803             426             MOV     DX,3B8H
E0C9 EE                 427             OUT     DX,AL                   ;DISABLE B/W VIDEO,EN HIGH RES
E0CA B800F0     R       428             MOV     AX,CODE                 ;SETUP SS SEG REG
E0CD 8ED0               429             MOV     SS,AX
E0CF B80CE0             430             MOV     DX,0E00CH               ;SETUP STARTING ROS ADDR
E0D2 BC16E0     P       431             MOV     SP,OFFSET C1            ;SETUP RETURN ADDRESS
E0D5 E93301             432             JMP     POS_CHECKSUM
E0D8 7505               433     C11:    JNE     ERR01                   ;HALT SYSTEM IF ERROR
                        434     ;-----------------------------------------
                        435     ;TEST.03
                        436     ;       8237 DMA INITIALIZATION CHANNEL REGISTER TEST
                        437     ;DESCRIPTION
                        438     ;       DISABLE THE 8237 DMA CONTROLLER.  VERIFY THAT TIMER 1 FUNCTIONS OK.
                        439     ;       WRITE/READ THE CURRENT ADDRESS AND WORD COUNT REGISTERS FOR ALL
                        440     ;       CHANNELS.  INITIALIZE AND START DMA FOR MEMORY REFRESH.
                        441     ;-----------------------------------------
                        442     ;       DISABLE DMA CONTROLLER
                        443
E0DA B004               444             MOV     AL,04                   ;DISABLE DMA CONTROLLER
E0DC E608               445             OUT     DMA08,AL
                        446
                        447     ;       VERIFY THAT TIMER 1 FUNCTIONS OK
                        448
E0DE B054               449             MOV     AL,54H                  ;SEL TIMER 1,LSB,MODE 2
E0E0 E643               450             OUT     TIMER+3,AL
E0E2 2BC9               451             SUB     CX,CX                   ;
E0E4 8AD9               452             MOV     BL,CL
E0E6 8AC1               453             MOV     AL,CL                   ;SET INITIAL TIMER CNT TO 0
E0E8 E641               454             OUT     TIMER+1,AL
E0EA                    455     CI2:                                    ; TIMER1_BITS_ON
E0EA B040               456             MOV     AL,40H                  ;LATCH TIMER 1 COUNT
E0EC E643               457             OUT     TIMER+3,AL
E0EE E441               458             IN      AL,TIMER+1              ;READ TIMER I COUNT
E0F0 0AD8               459             OR      BL,AL                   ;ALL BITS ON IN TIMER
E0F2 80FBFF             460             CMP     BL,0FFH                 ;YES - SEE IF ALL BITS GO OFF
E0F5 7404               461             JE      C13                     ; TIMER1_BITS_OFF
E0F7 E2F1               462             LOOP    CI2                     ; TIMER1_BITS_ON
E0F9 EBB4               463             JMP     SHORT ERR01             ;TIMER 1 FAILURE, HALT SYS
E0FB                    464     C13:                                    ; TIMER1_BITS_OFF
E0FB 8AC3               465             MOV     AL,BL                   ;SET TIMER 1 CNT
E0FD 2BC9               466             SUB     CX,CX
E0FF E641               467             OUT     TIMER+1,AL
E101                    468     CI4:                                    ; TIMER_LOOP
E101 B040               469             MOV     AL,40H                  ;LATCH TIMER 1 COUNT
E103 E643               470             OUT     TIMER+3,AL
E105 E441               471             IN      AL,TIMER+1              ;READ TIMER I COUNT
E107 22D8               472             AND     BL,AL
E109 7404               473             JZ      C15                     ; WRAP_DMA_REG
E10B E2F4               474             LOOP    C14                     ; TIMER_LOOP
```



```
LOC OBJ                 LINE    SOURCE

E10D EBA0               475             JMP     SHORT ERR01
                        476
                        477     ;       INITIALIZE TIMER 1 TO REFRESH MEMORY
                        478
E10F                    479     C15:                                    ; WRAP_DMA_REG
E10F B054               480             MOV     AL,54H                  ;SEL TIM 1, LSB, MODE 2
E111 E643               481             OUT     TIMER+3,AL              ;WRITE TIMER MODE REG
E113 B012               482             MOV     AL,18                   ;SETUP DIVISOR FOR REFRESH
E115 E641               483             OUT     TIMER+1,AL              ;WRITE TIMER 1 CNT REG
E117 E600               484             OUT     DMA+0DH,AL              ;SEND MASTER CLEAR TO DMA
                        485
                        486     ;       WRAP DMA CHANNELS ADDRESS AND COUNT REGISTERS
                        487
E119 B0FF               488             MOV     AL,0FFH                 ;WRITE PATTERN FFH TO ALL REGS
E11B 8AD8               489     C16:    MOV     BL,AL                   ;SAVE PATTERN FOR COMPARE
E11D 8AF8               490             MOV     BH,AL
E11F B90800             491             MOV     CX,8                    ;SETUP LOOP CNT
E122 BA0000             492             MOV     DX,DMA                  ;SETUP I/O PORT ADDR OF REG
E125 EE                 493     C17:    OUT     DX,AL                   ;WRITE PATTERN TO REG, LSB
E126 EE                 494             OUT     DX,AL                   ;MSB OF 16 BIT REG
E127 B80101             495             MOV     AX,0101H                ;AX TO ANOTHER PAT BEFORE RD
E12A EC                 496             IN      AL,DX                   ;READ 16-BIT DMA CH REG, LSB
E12B 8AE0               497             MOV     AH,AL                   ;SAVE LSB OF 16-BIT REG
E12D EC                 498             IN      AL,DX                   ;READ MSB OF DMA CH REG
E12E 3BD8               499             CMP     BX,AX                   ;PATTERN READ AS WRITTEN?
E130 7403               500             JE      C18                     ;YES - CHECK NEXT REG
E132 E97AFF             501             JMP     ERR01                   ;NO - HALT THE SYSTEM
E135                    502     C18:                                    ; NXT_DMA_CH
E135 42                 503             INC     DX                      ;SET I/O PORT TO NEXT CH REG
E136 E2E0               504             LOOP    C17                     ;WRITE PATTERN TO NEXT REG
E138 F6D0               505             NOT     AL                      ;SET PATTERH TO ZERO
E13A 74DF               506             JZ      C16                     ;WRITE TO CHANNEL REGS
                        507
                        508     ;       INITIALIZE AND START DMA FOR MEMORY REFRESH.
                        509
E13C B0FF               510             MOV     AL,0FFH                 ;SET CNT OF 64K FOR RAM REFRESH
E13E E601               511             OUT     DMA+1,AL
E140 E601               512             OUT     DMA+1,AL
E142 B058               513             MOV     AL,058H                 ;SET DMA MODE,CH 0,READ,AUOTINT
E144 E60B               514             OUT     DMA+0BH,AL              ;WRITE DMA MODE REG
E146 B000               515             MOV     AL,0                    ;ENABLE DMA CONTROLLER
E148 E608               516             OUT     DMA+8,AL                ;SETUP DMA COMMAND REG
E14A E60A               517             OUT     DMA+10,AL               ;ENABLE CHANNEL 0 FOR REFRESH
E14C B041               518             MOV     AL,41H                  ;SET MODE FOR CHANNEL 1
E14E E60B               519             OUT     DMA+0BH,AL
E150 B042               520             MOV     AL,42H                  ;SET MODE FOR CHANNEL 2
E152 E60B               521             OUT     DMA+0BH,AL              ;
E154 B043               522             MOV     AL,43H                  ;SET MODE FOR CHANNEL 3
E156 E60B               523             OUT     DMA+0BH,AL
                        524     ;-----------------------------------------
                        525     ;TEST.04
                        526     ;       BASE 16K READ/WRITE STORAGE TEST
                        527     ;
                        528     ;DESCRIPTION
                        529     ;       WRITE/READ/VERIFT DATA PATTERNS FF,55,AA,01, AND 00 TO 1ST 16K OF
                        530     ;       STORAGE.  VERIFT STORAGE ADDRESSABILITT.
                        531     ;       INITIALIZE THE 8259 INTERRUPT CONTROLLER CHIP FOR CHECKING
                        532     ;       MANUFACTURING TEST 2 MODE.
                        533     ;-----------------------------------------
                        534     ;       DETERMINE MEMORT SIZE AND FILL MEMORT WITH DATA
                        535
E158 B84000     R       536             MOV     AX,DATA                 ;POINT DS TO DATA SEG
E15B 8ED8               537             MOV     DS,AX                   ;
E15D 8B1E7200   R       538             MOV     BX,RESET_FLAG           ;SAVE  RESET_FLAG' IN BX
E161 2BC0               539             SUB     AX,AX                   ;SET ES AND DS TO 0
E163 8EC0               540             MOV     ES,AX                   ;SETUP ES SEGMENT REG
E165 8ED8               541             MOV     DS,AX
E167 2BFF               542             SUB     DI,DI
E169 E460               543             IN      AL,PORT_A               ;DETERMINE BASE RAM SIZE
E16B 240C               544             AND     AL,0CH                  ;ISOLATE RAM SIZE SWS
E16D 0404               545             ADD     AL, 4                   ; CALCULATE MEMORY SIZE
E16F B10C               546             MOV     CL, 12
E171 D3E0               547             SHL     AX, CL
E173 8BC8               548             MOV     CX, AX
E175 8AE0               549             MOV     AH, AL
E177 FC                 550             CLD                             ;SET DIR FLAG TO INCR
E178 AA                 551     C19:    STOSB                           ;FILL BASE RAM WITH DATA
```

```
LOC OBJ                 LINE    SOURCE


E179 E2FD                462            LOOP    C19                             ; LOOP TIL ALL ZERO
                         463
                         464    ;       DETERMINE IO CHANNEL RAM SIZE
                         465
E17B E462                466            IN      AL,PORT_C
E17D 240F                467            AND     AL,0FH
E17F 7418                468            JZ      C21
E181 BA0010              469            MOV     DX,1000H                        ; SEGMENT FOR I/O RAM
E184 8AE0                470            MOV     AH,AL
E186 B000                471            MOV     AL,0
E188                     472    C20:                                            ; FILL_IO:
E188 8EC2                473            MOV     ES,DX
E18A B90080              474            MOV     CX,8000H                        ; FILL 32K BYTES
E18D 2BFF                475            SUB     DI,DI
E18F F3                  476            REP     STOSB
E190 AA
E191 81C20008            477            ADD     DX,800H                         ; NEXT SEGMENT VALUE
E195 FECC                478            DEC     AH
E197 75EF                479            JNZ     C20                             ; FILL_IO
                         480    ;---------------------------------------
                         481    ;       INITIALIZE THE 8259 INTERRUPT CONTROLLER CHIP
                         482    ;---------------------------------------
E199                     483    C21:
E199 B013                484            MOV     AL,13H                          ;ICW1 - EDGE, SNGL, ICW4
E19B E620                485            OUT     INTA00,AL
E19D B008                486            MOV     AL,8                            ;SETUP ICW2 - INT TYPE 8 (8-F)
E19F E621                487            OUT     INTA01,AL
E1A1 B009                488            MOV     AL,9                            ;SETUP ICW4 - BUFFRD,8086 MODE
E1A3 E621                489            OUT     INTA01,AL
E1A5 2BC0                490            SUB     AX,AX                           ;POINT DS AND ES TO BEGIN
E1A7 8EC0                491            MOV     ES,AX                           ;  OF R/W STORAGE
E1A9 8E4000        R     492            MOV     SI,DATA                         ;POINT DS TO DATA SEG
E1AC 8ED8                493            MOV     DS,SI                           ;
E1AE 891E7200      R     494            MOV     RESET_FLAG,BX                   ;RESTORE RESET_FLAG
E1B2 813E72003412  R     495            CMP     RESET_FLAG,1234H                ;RESET_FLAG SET?
E1B8 7438                496            JE      C25                             ;YES - SKIP STG TEST
E1BA 8ED8                497            MOV     DS,AX                           ;POINT DS TO 1ST 16K OF STG
                         498    ;---------------------------------------
                         499    ;       CHECK FOR MANUFACTURING TEST 2 TO LOAD TEST PROGRAMS FROM KEYBOARD.
                         500    ;---------------------------------------
E1BC BCF03F              501            MOV     SP, 3FF0H                       ; ESTABLISH TEMPORART STACK
E1BF 8ED0                502            MOV     SS, AX
E1C1 8BF8                503            MOV     DI, AX
E1C3 BB2400              504            MOV     BX, 24H
E1C6 C706B6E2      R     505            MOV     WORD PTR [BX],OFFSET D11        ;SET UP KB INTERRUPT
E1CA 43                  506            INC     BX
E1CB 43                  507            INC     BX
E1CC 8C0F                508            MOV     [BX],CS
E1CE E88704              509            CALL    KBD_RESET               ; READ IN KB RESET CODE TO BL
E1D1 80FB65              510            CMP     BL,065H                 ; IS THIS MANUFACTURING TEST 2?
E1D4 750E                511            JNZ     C23                     ; JUMP IF NOT MAN. TEST
E1D6 B2FF                512            MOV     DL,255                  ; READ IN TEST PROGRAM
E1D8 E88A04              513    C22:    CALL    SP_TEST
E1DB 8AC3                514            MOV     AL,BL
E1DD AA                  515            STOSB
E1DE FECA                516            DEC     DL
E1E0 75F6                517            JNZ     C22                     ; JUMP IF NOT DONE YET
E1E2 CD3E                518            INT     3EH                     ;SET INTERUPT TYPE 62 ADDRESS F8H
E1E4                     519    C23:                                    ;CONTINUE IN NORMAL MODE
E1E4 0E                  520            PUSH    CS                      ; PUT SS BACK
E1E5 17                  521            POP     SS
E1E6 FA                  522            CLI                             ;
E1E7 BC18E0        R     523            MOV     SP,OFFSET C2                    ;SETUP RETURN ADDRESS
E1EA E920FE              524            JMP     STGTST                          ;GO TO RD/WRT STG SUBROUTINE
E1ED 7403                525    C24:    JE      C25                             ;GO TO NEXT TEST IF OK
E1EF E980FE              526            JMP     ERR01
                         527
                         528    ;       SETUP STACK SEG AND SP
                         529
E1F2                     530    C25:
E1F2 B83000              531            MOV     AX,STACK                        ; GET STACK VALUE
E1F5 8ED0                532            MOV     SS,AX                           ; SET THE STACK UP
E1F7 BC0001              533            MOV     SP,OFFSET TOS                   ; STACK IS READY TO GO
                         534
                         535    ;       SETUP THE NMI INTERRUPT VECTOR POINTER
                         536
```



```
LOC OBJ                 LINE    SOURCE


E1FA 26C7060800C3E2 R    537            MOV     ES:NMI_PTR,OFFSET NMI_INT
E201 26C7060A0000F0 R    538            MOV     ES:NMI_PTR+2,CODE
E208 E92A00              539            JMP     TST6                            ;GO TO NEXT TEST
                         540
E20B                     541    ROS_CHECKSUM    PROC    NEAR                    ; NEXT_ROS_MODULE
E20B B90020              542            MOV     CX,8192                         ;NUMBER OF BYTES TO ADD
E20E 32C0                543            XOR     AL,AL
E210                     544    C26:
E210 2E0207              545            ADD     AL,CS:[BX]                      ;
E213 43                  546            INC     BX                              ;POINT TO NEXT BYTE
E214 E2FA                547            LOOP    C26                             ;ADD ALL BYTES IN ROS MODULE
E216 0AC0                548            OR      AL,AL                           ;SUM = 0?
E218 C3                  549            RET
                         550    ROS_CHECKSUM    ENDP
                         551    ;---------------------------------------
                         552    ; INITIAL RELIABILITY TEST -- PHASE 2
                         553    ;---------------------------------------
                         554            ASSUME  CS:CODE,ES:ABS0
                         555
E219 50415249545920      556    D1      DB      'PARITY CHECK 2'
     4348454342032
  000E                   557    D1L     EQU     $-D1
E227 50415249545920      558    D2      DB      'PARITY CHECK 1'
     4348454342031
  000E                   559    D2L     EQU     $-D2
                         560    ;---------------------------------------
                         561    ;TEST.06
                         562    ;       8259 INTERRUPT CONTROLLER TEST
                         563    ;DESCRIPTION
                         564    ;       READ/WRITE THE INTERRUPT MASK REGISTER (IMR) WITH ALL ONES AND ZEROES.
                         565    ;       ENABLE SYSTEM INTERRUPTS.  MASK DEVICE INTERRUPTS OFF.  CHECK FOR
                         566    ;       HOT INTERRUPTS (UNEXPECTED).
                         567    ;---------------------------------------
E235                     568    TST6:
E235 2BC0                569            SUB     AX,AX                           ;SET UP ES REG
E237 8EC0                570            MOV     ES,AX
                         571
                         572    ;------ SET UP THE INTERRUPT 5 POINTER TO A DUMMY
                         573
E239 26C706140054FF R    574            MOV     ES:INT5_PTR,OFFSET PRINT_SCREEN         ;PRINT SCREEN
E240 26C706160000F0 R    575            MOV     ES:INT5_PTR+2,CODE                      ;
                         576
                         577    ;       TEST THE IMR REGISTER
                         578
E247 FA                  579            CLI                                     ;DISABLE INTERRUPTS
E248 B000                580            MOV     AL,0                            ;SET IMR TO ZERO
E24A E621                581            OUT     INTA01,AL
E24C E421                582            IN      AL,INTA01                       ;READ IMR
E24E 0AC0                583            OR      AL,AL                           ;IMR = 0?
E250 752B                584            JNZ     D6                              ;GO TO ERR ROUTINE IF NOT 0
E252 B0FF                585            MOV     AL,0FFH                         ;DISABLE DEVICE INTERRUPTS
E254 E621                586            OUT     INTA01,AL                       ;WRITE TO IMR
E256 E421                587            IN      AL,INTA01                       ;READ IMR
E258 0401                588            ADD     AL,1                            ;ALL IMR BIT ON?
E25A 7521                589            JNZ     D6                              ;NO - GO TO ERR ROUTINE
                         590
                         591    ;       CHECK FOR HOT INTERRUPTS
                         592
E25C FC                  593            CLD                                     ;SET DIR FLAG TO GO FORWARD
E25D B90800              594            MOV     CX,8                            ;SETUP TEMP INT RTNE IN PRT TBL
E260 BF2000              595            MOV     DI,OFFSET INT_PTR               ;GET ADDRESS OF INT PROC TABLE
E263                     596    D3:                                             ; VECTBL0:
E263 8B86E2        R     597            MOV     AX,OFFSET D11                   ;MOVE ADDR OF INTR PROC TO TBL
E266 AB                  598            STOSW
E267 8B00F0        R     599            MOV     AX,CODE                         ;GET ADDR OF INTR PROC SEG
E26A AB                  600            STOSW
E26B 83C304              601            ADD     BX,4                            ;SET BX TO POINT TO NEXT VAL
E26E E2F3                602            LOOP    D3                              ; VECTBL0
                         603
                         604    ;       INTERRUPTS ARE MASKED OFF.  CHECK THAT NO INTERRUPTS OCCUR.
                         605
E270 32E4                606            XOR     AH,AH                           ;CLEAR AN REG
E272 FB                  607            STI                                     ;ENABLE EXTERNAL INTERRUPTS
E273 2BC9                608            SUB     CX,CX                           ;WAIT 1 SEC FOR ANY INTRS THAT
E275 E2FE                609    D4:     LOOP    D4                              ;MIGHT OCCUR
E277 E2FE                610    D5:     LOOP    D5
E279 0AE4                611            OR      AH,AH                           ;DID ANY INTERRUPTS OCCUR?
```

```
LOC OBJ                 LINE    SOURCE

E27B 7408               612             JZ      07                      ;NO - GO TO NEXT TEST
E27D BA0101             613     D6:     MOV     DX,101H                 ;BEEP SPEAKER IF ERROR
E280 E8AD03             614             CALL    ERR_BEEP                ;GO TO BEEP SUBROUTINE
E283 FA                 615             CLI
E284 F4                 616             HLT                             ;HALT THE SYSTEM
                        617     ;-----------------------------------------
                        618     ;TEST.7
                        619     ;       8253 TIMER CHECKOUT
                        620     ;DESCRIPTION
                        621     ;       VERIFY THAT THE SYSTEM TIMER (0) DOESN'T COUNT TOO FAST NOR TOO
                        622     ;       SLOW.
                        623     ;-----------------------------------------
E285                    624     07:
E285 B400               625             MOV     AH,0                    ;RESET TIMER INTR RECVD FLAG
E287 32ED               626             XOR     CH,CH                   ;CLEAR THE CH REG
E289 B0FE               627             MOV     AL,0FEH                 ;MASK ALL INTRS EXCEPT LVL 0
E28B E621               628             OUT     INTA01,AL               ;WRITE THE 8259 IMR
E28D B010               629             MOV     AL,00010000B            ;SEL TIM 0, LSB, MODE 0, BINART
E28F E643               630             OUT     TIM_CTL,AL              ;WRITE TIMER CONTROL MODE REG
E291 B116               631             MOV     CL,16H                  ;SET PGM LOOP CNT
E293 8AC1               632             MOV     AL,CL                   ;SET TIMER 0 CNT REG
E295 E640               633             OUT     TIMER0,AL               ;WRITE TIMER 0 CNT REG
E297 F6C4FF             634     08:     TEST    AH,0FFH                 ;DID TIMER 0 INTERRUPT OCCUR?
E29A 7504               635             JNZ     09                      ; YES - CHECK TIMER OP FOR SLOW TIME
E29C E2F9               636             LOOP    08                      ;WAIT FOR INTR FOR SPECIFIED TIME
E29E EBDD               637             JMP     D6                      ;TIMER 0 INTR DIDN T OCCUR - ERR
E2A0 B112               638     09:     MOV     CL,18                   ;SET PGM LOOP CNT
E2A2 B0FF               639             MOV     AL,0FFH                 ;WRITE TIMER 0 CNT REG
E2A4 E640               640             OUT     TIMER0,AL
E2A6 B400               641             MOV     AH,0                    ;RESET INTR RECEIVED FLAG
E2A8 B0FE               642             MOV     AL,0FEH                 ;REENABLE TIMER 0 INTERRUTS
E2AA E621               643             OUT     INTA01,AL
E2AC F6C4FF             644     010:    TEST    AH,0FFH                 ;DID TIMER 0 INTERRUPT OCCUR?
E2AF 75CC               645             JNZ     D6                      ;YES - TIMER CNTING TOO FAST, ERR
E2B1 E2F9               646             LOOP    010                     ;WAIT FOR INTR FOR SPECIFIED TIME
E2B3 E93600             647             JMP     TST8                    ;GO TO NEXT TEST ROUTINE
                        648     ;-----------------------------------------
                        649     ;       TEMPORARY INTERRUPT SERVICE ROUTINE
                        650     ;-----------------------------------------
E2B6                    651     D11     PROC    NEAR
E2B6 B401               652             MOV     AH,1
E2B8 50                 653             PUSH    AX                      ;SAVE REG AX CONTENTS
E2B9 B0FF               654             MOV     AL,0FFH                 ;MASK ALL INTERRUPTS OFF
E2BB E621               655             OUT     INTA01,AL
E2BD B020               656             MOV     AL,EOI
E2BF E620               657             OUT     INTA00,AL
E2C1 5B                 658             POP     AX                      ;RESTORE REG AX CONTENTS
E2C2 CF                 659             IRET
                        660     D11     ENDP
                        661
E2C3                    662     NMI_INT PROC    NEAR
E2C3 50                 663             PUSH    AX                      ;SAVE ORIG CONTENTS OF AX
E2C4 E462               664             IN      AL,PORT_C
E2C6 A840               665             TEST    AL,40H                  ;IO CH PARITY CHECK?
E2C8 7408               666             JZ      D12                     ;YES - FLAG IS SET TO 0
E2CA BE19E2         R   667             MOV     SI,OFFSET D1            ;ADDR OF ERROR MSG
E2CD B90E00             668             MOV     CX,D1L                  ;MSG LENGTH
E2D0 EB0A               669             JMP     SHORT D13               ;DISPLAY ERROR MSG
E2D2                    670     D12:
E2D2 A880               671             TEST    AL,80H                  ;PLANAR RAM P-CHECK?
E2D4 7410               672             JZ      D14                     ;NO - AUX INT
E2D6 BE27E2         R   673             MOV     SI,OFFSET D2            ;ADDR OF ERROR MSG
E2D9 B90E00             674             MOV     CX,D2L                  ;MSG LENGTH
E2DC                    675     D13:
E2DC B80000             676             MOV     AX,0                    ;INIT AND SET MODE FOR VIDEO
E2DF CD10               677             INT     10H                     ;CALL VIDEO_IO PROCEDURE
E2E1 E8E603             678             CALL    P_MSG                   ;PRINT ERROR MSG
E2E4 FA                 679             CLI
E2E5 F4                 680             HLT                             ;HALT SYSTEM
E2E6                    681     D14:
E2E6 58                 682             POP     AX                      ;RESTORE ORIG CONTENTS OF AX
E2E7 CF                 683             IRET
                        684     NMI_INT ENDP
```



```
LOC OBJ                 LINE    SOURCE

                        685     ;-----------------------------------
                        686     ; INITIAL RELIABILITY TEST -- PHASE 3
                        687     ;-----------------------------------
                        688             ASSUME  CS:CODE,DS:DATA
                        689
E2E8 20323031           690     E1      DB      ' 201'
  0004                  691     E1L     EQU     $-E1
                        692
                        693     ;       ESTABLISH BIOS SUBROUTINE CALL INTERRUPT VECTORS
                        694
E2EC                    695     TST8:
E2EC FC                 696             CLD                             ;SET DIR FLAG TO GO FORWARD
E2ED BF4000             697             MOV     DI,OFFSET VIDEO_INT     ;SETUP ADDR TO INTR AREA
E2F0 0E                 698             PUSH    CS
E2F1 1F                 699             POP     DS                      ;SETUP ADDR OF VECTOR TABLE
E2F2 BE13FF             700             MOV     SI,0FF13H               ; OFFSET VECTOR_TABLE+32
E2F5 B92000             701             MOV     CX,20H
E2F8 F3                 702             REP     MOVSW                   ;MOVE VECTOR TABLE TO RAM
E2F9 A5
                        703
                        704     ;       SETUP TIMER 0 TO MODE 3
                        705
E2FA B0FF               706             MOV     AL,0FFH                 ;DISABLE ALL DEVICE INTERRUPTS
E2FC E621               707             OUT     INTA01,AL
E2FE B036               708             MOV     AL,36H                  ;SEL TIM 0,LSB,MSB,MODE 3
E300 E643               709             OUT     TIMER+3,AL              ;WRITE TIMER MODE REG
E302 B000               710             MOV     AL,0
E304 E640               711             OUT     TIMER,AL                ;WRITE LSB TO TIMER 0 REG
E306 E640               712             OUT     TIMER,AL                ;WRITE MSB TO TIMER 0 REG
                        713
                        714     ;       SETUP TIMER 0 TO BLINK LED IF MANUFACTURING TEST MODE
                        715
                        716             ASSUME  DS:DATA
E308 B84000         R   717             MOV     AX,DATA                 ;POINT DS TO DATA SEG
E30B 8ED8               718             MOV     DS,AX
E30D E87803             719             CALL    KBD_RESET               ;SEND SOFTWARE RESET TO KEYBRD
E310 80FBAA             720             CMP     BL,0AAH                 ;SCAN CODE  AA' RETURNED?
E313 7426               721             JE      E3                      ;YES - CONTINUE (NON MFG MODE)
E315 B03C               722             MOV     AL,3CH                  ;EN KBD, SET KBD CLK LINE LOW
E317 E661               723             OUT     PORT_B,AL               ;WRITE 8255 PORT B
E319 90                 724             NOP
E31A 90                 725             NOP
E31B E460               726             IN      AL,PORT_A               ;WAS A BIT CLOCKED IN?
E31D 24FF               727             AND     AL,0FFH
E31F 7516               728             JNZ     E2                      ;YES - CONTINUE (NON MFG MODE)
E321 FE061200       R   729             INC     MFG_TST                 ;ELSE SET SW FOR MFG TEST MODE
E325 26C7062000B2E6 R   730             MOV     ES:INT_ADDR,OFFSET BLINK_INT    ;SETUP TIMER INTR TO BLINK LED
E32C 26C706220000F0 R   731             MOV     ES:INT_ADDR+2,CODE
E333 B0FE               732             MOV     AL,0FEH                 ;ENABLE TIMER INTERRUPT
E335 E621               733             OUT     INTA01,AL
E337                    734     E2:                                     ; JUMPER_NOT_IN:
E337 B0CC               735             MOV     AL,0CCH                 ;RESET THE KEYBOARD
E339 E661               736             OUT     PORT_B,AL
                        737     ;-----------------------------------------
                        738     ;TEST.05
                        739     ;       ROS CHECKSUM II
                        740     ;DESCRIPTION
                        741     ;       A CHECKSUM IS DONE FOR THE 4 ROS MODULES CONTAINING BASIC CODE
                        742     ;-----------------------------------------
E33B                    743     E3:
E33B B204               744             MOV     DL,4                    ;NO. OF ROS MODULES TO CHECK
E33D BB0060             745             MOV     BX,6000H                ;SETUP STARTING ROS ADDR
E340                    746     E4:                                     ; CHECK_ROS:
E340 E8C8FE             747             CALL    ROS_CHECKSUM
E343 7507               748             JNE     E5                      ;BEEP SPEAKER IF ERROR
E345 FECA               749             DEC     DL                      ;ANY MORE TO DO?
E347 75F7               750             JNZ     E4                      ;YES - CONTINUE
E349 EB0790             751             JMP     E6                      ;NO - GO TO NEXT TEST
E34C                    752     E5:                                     ; ROS_ERROR:
E34C BA0101             753             MOV     DX,101H
E34F E8DE02             754             CALL    ERR_BEEP                ;BEEP SPEAKER
```

```
LOC OBJ                 LINE    SOURCE

                        755     ;----------------------------------------
                        756     ;TEST.08
                        757     ;       INITIALIZE AND START CRT CONTROLLER (6845)
                        758     ;       TEST VIDEO READ/WRITE STORAGE.
                        759     ;DESCRIPTION
                        760     ;       RESET THE VIDEO ENABLE SIGNAL.
                        761     ;       SELECT ALPHANUMERIC MODE, 40 * 25, B & W.
                        762     ;       READ/WRITE DATA PATTERNS TO STG. CHECK STG ADDRESSABILITY.
                        763     ;----------------------------------------
E352                    764     E6:
E352 E460               765             IN      AL,PORT_A               ;READ SENSE SWITCHES
E354 B400               766             MOV     AH,0
E356 A31000         R   767             MOV     EQUIP_FLAG,AX           ;STORE SENSE SW INFO
E359 2430               768             AND     AL,30H                  ;ISOLATE VIDEO SWS
E35B 7503               769             JNZ     E7                      ;VIDEO SWS SET TO 0?
E35D E99800             770             JMP     E19                     ;SKIP VIDEO TESTS FOR BURN-IN
E360                    771     E7:                                     ; TEST_VIDEO:
E360 86E0               772             XCHG    AH,AL
E362 80FC30             773             CMP     AH,30H                  ;B/W CARD ATTACHED?
E365 7409               774             JE      E8                      ;YES - SET MODE FOR B/W CARD
E367 FEC0               775             INC     AL                      ;SET COLOR MODE FOR CULOR CD
E369 80FC20             776             CMP     AH,20H                  ;80X25 MODE SELECTED?
E36C 7502               777             JNE     E8                      ;NO - SET MODE FOR 40X25
E36E B003               778             MOV     AL,3                    ;SET MODE FOR 80X25
E370                    779     E8:                                     ; SET_MODE:
E370 50                 780             PUSH    AX                      ;SAVE VIDEO MODE ON STACK
E371 2AE4               781             SUB     AH,AH                   ;INITIALIZE TO ALPHANUMERIC MD
E373 CD10               782             INT     10H                     ;CALL VIDEO_IO
E375 58                 783             POP     AX                      ;RESTORE VIDEO SENSE SWS IH AH
E376 50                 784             PUSH    AX                      ; RESAVE VALUE
E377 8B00B0             785             MOV     BX,0B000H               ;BEG VIDEO RAM ADDR B/W CD
E37A BAB803             786             MOV     DX,3B8H                 ;MODE REG FOR B/W
E37D B90010             787             MOV     CX,4096                 ;RAM BYTE CNT FOR B/W CD
E380 B001               788             MOV     AL,1                    ; SET MODE FOR BW CARD
E382 80FC30             789             CMP     AH,30H                  ;B/W VIDEO CARD ATTACHED?
E385 740B               790             JE      E9                      ;YES - GO YEST VIDEO SYG
E387 BB0088             791             MOV     BX,0B800H               ;BEG VIDEO RAM ADDR COLOR CD
E38A BAD803             792             MOV     DX,3D8H                 ;MODE REG FOR COLOR CD
E38D B90040             793             MOV     CX,4000H                ;RAM BYTE CNT FOR COLOR CD
E390 FEC8               794             DEC     AL                      ; SET MODE TO 0 FOR COLOR CD
E392                    795     E9:                                     ; TEST_VIDEO_STG:
E392 EE                 796             OUT     DX,AL                   ;DISABLE VIDEO FOR COLOR CD
E393 8EC3               797             MOV     ES,BX                   ;POINT ES TO VIDEO RAM STG
E395 B84000         R   798             MOV     AX,DATA                 ;POINT DS TO DATA SEGMENT
E398 8ED8               799             MOV     DS,AX
E39A 813E72003412   R   800             CMP     RESET_FLAG,1234H        ;POD INITIATED BY KBD RESET?
E3A0 740D               801             JE      E10                     ;YES - SKIP VIDEO RAM TEST
E3A2 8EDB               802             MOV     DS,BX                   ;POINT DS TO VIDEO RAM STG
E3A4 E876FC             803             CALL    STGTST_CNT              ;GO TEST VIDEO R/W STG
E3A7 7406               804             JE      E10                     ;STG OK - CONTINUE TESTING
E3A9 BA0201             805             MOV     DX,102H                 ;SETUP # OF BEEPS
E3AC E88102             806             CALL    ERR_BEEP                ;GO BEEP SPEAKER
                        807     ;----------------------------------------
                        808     ;TEST.09
                        809     ;       SETUP VIDEO DATA ON SCREEH FOR VIDEO LINE TEST.
                        810     ;DESCRIPTION
                        811     ;       ENABLE VIDEO SIGNAL AND SET MODE.
                        812     ;       DISPLAT A HORIZONTAL BAR ON SCREEN.
                        813     ;----------------------------------------
E3AF                    814     E10:
E3AF 58                 815             POP     AX                      ;GET VIDEO SENSE SWS (AH)
E3B0 50                 816             PUSH    AX                      ;SAVE IT
E3B1 B400               817             MOV     AH,0                    ;ENABLE VIDEO AND SET MODE
E3B3 CD10               818             INT     10H                     ; VIDEO
E3B5 B82070             819             MOV     AX,7020H                ;WRT BLANKS IN REVERSE VIDEO
E3B8 2BFF               820             SUB     DI,DI                   ;SETUP STARTING LOC
E3BA B92800             821             MOV     CX,40                   ;NO. OF BLANKS TO DISPLAY
E3BD FC                 822             CLD                             ;SET DIR FLAG TO INCREMENT
E3BE F3                 823             REP     STOSW                   ;WRITE VIDEO STORAGE
E3BF AB
```



```
LOC OBJ                 LINE    SOURCE

                        824     ;----------------------------------------
                        825     ;TEST.10
                        826     ;       CRT INTERFACE LINES TEST
                        827     ;DESCRIPTION
                        828     ;       SENSE ON/OFF TRANSITION OF THE VIDEO ENABLE AND HORIZONTAL
                        829     ;       SYNC LINES.
                        830     ;----------------------------------------
E3C0 58                 831             POP     AX                      ;GET VIDEO SENSE SW INFO
E3C1 50                 832             PUSH    AX                      ;SAVE IT
E3C2 80FC30             833             CMP     AH,30H                  ;B/W CARD ATTACHED?
E3C5 8ABA03             834             MOV     DX,03BAH                ;SETUP ADDR OF BW STATUS PORT
E3C8 7403               835             JE      E11                     ;YES - GO TEST LINES
E3CA BADA03             836             MOV     DX,03DAH                ;COLOR CARD IS ATTACHED
E3CD                    837     E11:                                    ; LINE_TST:
E3CD B408               838             MOV     AH,8
E3CF                    839     E12:                                    ; OFLOOP_CNT:
E3CF 2BC9               840             SUB     CX,CX
E3D1 EC                 841     E13:    IN      AL,DX                   ;READ CRT STATUS PORT
E3D2 22C4               842             AND     AL,AH                   ;CHECK VIDEO/HORZ LINE
E3D4 7504               843             JNZ     E14                     ;ITS ON - CHECK IF IT GOES OFF
E3D6 E2F9               844             LOOP    E13                     ;LOOP TILL ON OR TIMEOUT
E3D8 EB13               845             JMP     SHORT E17               ;GO PRINT ERROR MSG
E3DA 2BC9               846     E14:    SUB     CX,CX
E3DC EC                 847     E15:    IN      AL,DX                   ;READ CRT STATUS PORT
E3DD 22C4               848             AND     AL,AH                   ;CHECK VIDEO/HORZ LINE
E3DF 7404               849             JZ      E16                     ;ITS ON - CHECK NEXT LINE
E3E1 E2F9               850             LOOP    E15                     ;LOOP IF OFF TILL IT GOES ON
E3E3 EB08               851             JMP     SHORT E17
E3E5                    852     E16:                                    ; NXT_LINE:
E3E5 B103               853             MOV     CL,3                    ;GET NEXT BIT TO CHECK
E3E7 D2EC               854             SHR     AH,CL                   ;
E3E9 75E4               855             JNZ     E12                     ;GO CHECK HORIZONTAL LINE
E3EB EB06               856             JMP     SHORT E18               ;DISPLAT CURSOR ON SCREEN
E3ED                    857     E17:                                    ; CRT_ERR:
E3ED BA0201             858             MOV     DX,102H
E3F0 E83D02             859             CALL    ERR_BEEP                ;GO BEEP SPEAKER
E3F3                    860     E18:                                    ; DISPLAT_CURSOR:
E3F3 58                 861             POP     AX                      ;GET VIDEO SENSE SWS (AH)
E3F4 B400               862             MOV     AH,0                    ;SET MODE AND DISPLAT CURSOR
E3F6 CD10               863             INT     10H                     ;CALL VIDEO I/O PROCEDURE
                        864     ;----------------------------------------
                        865     ;TEST.11
                        866     ;       ADDITIONAL READ/WRITE STORAGE TEST
                        867     ;DESCRIPTION
                        868     ;       WRITE/READ DATA PATTERNS TO ANY READ/WRITE STORAGE AFTER THE BASIC
                        869     ;       16K.  STORAGE ADDRESSABILITT IS CHECKED.
                        870     ;----------------------------------------
                        871             ASSUME  DS:DATA
E3F8                    872     E19:
E3F8 B84000         R   873             MOV     AX,DATA
E3FB 8ED8               874             MOV     DS,AX
                        875
                        876     ;       DETERMINE RAM SIZE ON PLANAR BOARD
                        877
E3FD 8A261000       R   878             MOV     AH,BYTE PTR EQUIP_FLAG  ;GET SENSE SWS INFO
E401 80E40C             879             AND     AH,0CH                  ;ISOLATE RAM SIZE SWS
E404 B004               880             MOV     AL,4
E406 F6E4               881             MUL     AH
E408 0410               882             ADD     AL,16                   ;ADD BASIC 16K
E40A 8BD0               883             MOV     DX,AX                   ;SAVE PLANAR RAM SIZE IN DX
E40C 8BD8               884             MOV     BX,AX                   ;  AND IN BX
                        885
                        886     ;       DETERMINE IO CHANNEL RAM SIZE
                        887
E40E E462               888             IN      AL,PORT_C               ;READ IO CH RAM SIZE SWS
E410 240F               889             AND     AL,0FH                  ;ISOLATE FROM OTHER BITS
E412 B420               890             MOV     AH,32
E414 F6E4               891             MUL     AH
E416 A31500         R   892             MOV     IO_RAM_SIZE,AX          ;SAVE IO CHANNEL RAM SIZE
E419 83FB40             893             CMP     BX,40H                  ;PLANAR RAM SIZE = 64K?
E41C 7402               894             JE      E20                     ;YES - ADD IO CHN RAM SIZE
E41E 2BC0               895             SUB     AX,AX                   ;NO - DON T ADD ANY IO RAM
E420                    896     E20:                                    ; ADD_IO_SIZE:
E420 03C3               897             ADD     AX,BX                   ;SUM TOTAL RAM SIZE
E422 A31300         R   898             MOV     MEMORY_SIZE,AX          ;SETUP MEMORY SIZE PARM
E425 813E72003412   R   899             CMP     RESET_FLAG,1234H        ;POD INITIATED BY KBD RESET?
```

```
LOC OBJ                 LINE    SOURCE

E428 7440               900             JE      E22                     ;YES - SKIP MEMORY TEST
                        901
                        902     ;       TEST ANY OTHER READ/WRITE STORAGE AVAILABLE
                        903
E420 B80004             904             MOV     BX,400H
E430 B91000             905             MOV     CX,16
E433                    906     E21:
E433 3BD1               907             CMP     DX,CX                   ;ANY MORE STG TO BE TESTED?
E435 7646               908             JBE     E23                     ;NO - GO TO NEXT TEST
E437 8EDB               909             MOV     DS,BX                   ;SETUP STG ADDR IN DS AND ES
E439 8EC3               910             MOV     ES,BX
E43B 83C110             911             ADD     CX,16                   ;INCREMENT STG BYTE COUNTER
E43E 81C30004           912             ADD     BX,400H                 ;SET POINTER TO NEXT 16K BLK
E442 51                 913             PUSH    CX                      ;SAVE REGS
E443 53                 914             PUSH    BX
E444 52                 915             PUSH    DX
E445 E802FB             916             CALL    STGTST                  ;GO TEST A 16K BLK OF STG
E448 5A                 917             POP     DX
E449 5B                 918             POP     BX                      ;RESTORE REGS
E44A 59                 919             POP     CX
E44B 74E6               920             JE      E21                     ;CHECK IF MORE STG TO TEST
                        921
                        922     ;       PRINT FAILING ADDRESS AND XOR'ED PATTERN IF DATA COMPARE ERROR
                        923
E44D 8CDA               924             MOV     DX,DS                   ;CONVERT FAILING HIGH-ORDER
E44F 8AE8               925             MOV     CH,AL                   ;SAVE FAILING BIT PATTERN
E451 8AC6               926             MOV     AL,DH                   ;GET FAILING ADDR (HIGH BYTE)
E453 B104               927             MOV     CL,4
E455 D2E8               928             SHR     AL,CL                   ;RIGHT-JUSTIFY HIGH BYTE
E457 E83E00             929             CALL    XLAT_PRINT_CODE         ;CONVERT AND PRINT CODE
E45A 8AC6               930             MOV     AL,DH
E45C 240F               931             AND     AL,0FH
E45E E83700             932             CALL    XLAT_PRINT_CODE         ;CONVERT AND PRINT CODE
E461 8AC5               933             MOV     AL,CH                   ;GET FAILING BIT PATTERN
E463 B104               934             MOV     CL,4                    ; AND ISOLATE LEFTMOST NIBBLE
E465 D2E8               935             SHR     AL,CL
E467 E82E00             936             CALL    XLAT_PRINT_CODE         ;CONVERT AND PRINT CODE
E46A 8AC5               937             MOV     AL,CH                   ;GET FAILING BIT PATTERN AND
E46C 240F               938             AND     AL,0FH                  ;  ISOLATE RIGHTMOST NIBBLE
E46E E82700             939             CALL    XLAT_PRINT_CODE         ;CONVERT AND PRINT CODE
E471 BEE8E2       R     940             MOV     SI,OFFSET E1            ;SETUP ADDRESS OF ERROR MSG
E474 B90400             941             MOV     CX,E1L                  ;GET MSG BYTE COUNT
E477 E85002             942             CALL    P_MSG                   ;PRINT ERROR MSG
E47A                    943     E22:                                    ; GO_TST12:
E47A E99A00             944             JMP     TST12                   ;GO TO NEXT TEST
E47D                    945     E23:                                    ; STG_TEST_DONE:
E47D B84000       R     946             MOV     AX,DATA                 ;POINT DS TO DATA SEGMENT
E480 8ED8               947             MOV     DS,AX                   ; CHG MADE 3/27/81
E482 8B161500     R     948             MOV     DX,IO_RAM_SIZE          ;GET IO CHANNEL RAM SIZE
E486 0BD2               949             OR      DX,DX                   ;SET FLAG RESULT
E488 74F0               950             JZ      E22                     ;NO IO RAM, GO TO NEXT TEST
E48A B90000             951             MOV     CX,0
E48D 81FB0010           952             CMP     BX,1000H                ;HAS IO RAM BEEN TESTED
E491 77E7               953             JA      E22                     ;YES - GO TO NEXT TEST
E493 BB0010             954             MOV     BX,1000H                ;SETUP BEG LOC FOR IO RAM
E496 EB9B               955             JMP     SHORT E21               ;GO TEST IO CHANNEL RAM
                        956     ;---------------------------------------
                        957     ;       CONVERT AND PRINT ASCII CODE
                        958     ;
                        959     ;       AL MUST CONTAIN NUMBER TO BE CONVERTED.
                        960     ;       AX AND BX DESTROYED.
                        961     ;---------------------------------------
E498                    962     XLAT_PRINT_CODE  PROC    NEAR
E498 1E                 963             PUSH    DS                      ;SAVE DS VALUE
E499 0E                 964             PUSH    CS                      ;POINT DS TO CODE SEG
E49A 1F                 965             POP     DS
E49B 8B87E4             966             MOV     BX,0E4B7H               ; OFFSET ASCII_TBL-XLAT TABLE
E49E D7                 967             XLATB
E49F B40E               968             MOV     AH,14
E4A1 B700               969             MOV     BH,0
E4A3 CD10               970             INT     10H                     ;CALL VIDEO_IO
E4A5 1F                 971             POP     DS                      ;RESTORE ORIG VALUE IN DS
E4A6 C3                 972             RET
                        973     XLAT_PRINT_CODE  ENDP
```



```
LOC OBJ                 LINE    SOURCE

                        974     ;---------------------------------------
                        975     ; INITIAL RELIABILITY TEST -- PHASE 4
                        976     ;---------------------------------------
                        977             ASSUME  CS:CODE,DS:DATA
E4A7 20333031           978     F1      DB      ' 301'
  0004                  979     F1L     EQU     $-F1            ; KEYBOARD MESSAGE
E4AB 313331             980     F2      DB      '131'
  0003                  981     F2L     EQU     $-F2            ; CASSETTE MESSAGE
E4AE 363031             982     F3      DB      '601'
  0003                  983     F3L     EQU     $-F3            ; DISKETTE MESSAGE
                        984
E4B1                    985     F4      LABEL   WORD            ; PRINTER SOURCE TABLE
E4B1 BC03               986             DW      3BCH
E4B3 7803               987             DW      378H
E4B5 7802               988             DW      278H
E4B7                    989     F4E     LABEL   WORD
E4B7 30313233343536     990     ASCII_TBL       DB      '0123456789ABCDEF'
     3738393A4243
     4546
                        991     ;---------------------------------------
                        992     ;TEST.12
                        993     ;       KEYBOARD TEST
                        994     ;DESCRIPTION
                        995     ;       RESET THE KEYBOARD AND CHECK THAT SCAN CODE  AA' IS RETURNED
                        996     ;       TO THE CPU.  CHECK FOR STUCK KEYS.
                        997     ;---------------------------------------
E4C7                    998     TST12:
E4C7 B84000       R     999             MOV     AX,DATA                 ;POINT DS TO DATA SEG
E4CA 8ED8              1000             MOV     DS,AX
E4CC 803E120001   R    1001             CMP     MFG_TST,1               ;MANUFACTURING TEST MODE?
E4D1 7439              1002             JE      F7                      ;YES - SKIP KEYBOARD TEST
E4D3 E8B201            1003             CALL    KBD_RESET               ;ISSUE SOFTWARE RESET TO KEYBRD
E4D6 E32B              1004             JCXZ    F6                      ;PRINT ERR MSG IF NO INTERRUPT
E4D8 B040              1005             MOV     AL,40H                  ;ENABLE KEYBOARD
E4DA E661              1006             OUT     PORT_B,AL
E4DC 80FBAA            1007             CMP     BL,0AAH                 ;SCAN CODE AS EXPECTED?
E4DF 7522              1008             JNE     F6                      ;NO - DISPLAY ERROR MSG
                       1009
                       1010     ;       CHECK FOR STUCK KEYS
                       1011
E4E1 B0CC              1012             MOV     AL,0CCH                 ;CLR KBD, SET CLK LINE HIGH
E4E3 E661              1013             OUT     PORT_B,AL
E4E5 B04C              1014             MOV     AL,4CH                  ;ENABLE KBD,CLK IN NEXT BYTE
E4E7 E661              1015             OUT     PORT_B,AL
E4E9 2BC9              1016             SUB     CX,CX
E4EB                   1017     F5:                                     ; KBD_WAIT:
E4EB E2FE              1018             LOOP    F5                      ;DELAY FOR A WHILE
E4ED E460              1019             IN      AL,KBD_IN               ;CHECK FOR STUCK KEYS
E4EF 3C00              1020             CMP     AL,0                    ;SCAN CODE = 0?
E4F1 7419              1021             JE      F7                      ;YES - CONTINUE TESTING
E4F3 8AE8              1022             MOV     CH,AL                   ;SAVE SCAN CODE
E4F5 B104              1023             MOV     CL,4
E4F7 D2E8              1024             SHR     AL,CL                   ;RIGHT-JUSTIFY HIGH BYTE
E4F9 E89CFF            1025             CALL    XLAT_PRINT_CODE         ;CONVERT AND PRINT
E4FC 8AC5              1026             MOV     AL,CH                   ;RECOVER SCAN CODE
E4FE 240F              1027             AND     AL,0FH                  ;ISOLATE LOW ORDER BYTE
E500 E895FF            1028             CALL    XLAT_PRINT_CODE         ;CONVERT AND PRINT
E503 BEA7E4       R    1029     F6:     MOV     SI,OFFSET F1            ;GET MSG ADDR
E506 B90400            1030             MOV     CX,F1L                  ;GET MSG BYTE COUNT
E509 E8BE01            1031             CALL    P_MSG                   ;PRINT MSG ON SCREEN
                       1032
                       1033     ;       SETUP INTERRUPT VECTOR TABLE
                       1034
E50C                   1035     F7:                                     ; SETUP_INT_TABLE:
E50C 2BC0              1036             SUB     AX,AX
E50E 8EC0              1037             MOV     ES,AX
E510 B93000            1038             MOV     CX,24*2                 ;GET VECTOR CNT
E513 0E                1039             PUSH    CS                      ;SETUP DS SEG REG
E514 1F                1040             POP     DS
E515 BEF3FE            1041             MOV     SI,0FEF3H               ; OFFSET VECTOR_TABLE
E518 BF2000            1042             MOV     DI,OFFSET INT_PTR
E51B FC                1043             CLD
E51C F3                1044             REP     MOVSW
E51D A5
```

```
LOC OBJ                 LINE    SOURCE

                        1045    ;--------------------------------------------
                        1046    ;TEST.13
                        1047    ;       CASSETTE DATA WRAP TEST
                        1048    ;DESCRIPTION
                        1049    ;       TURN CASSETTE MOTOR OFF. WRITE A BIT OUT TO THE CASSETTE DATA BUS.
                        1050    ;       VERIFY THAT CASSETTE DATA READ IS WITHIN A VALID RANGE.
                        1051    ;--------------------------------------------
                        1052
                        1053    ;       TURN THE CASSETTE MOTOR OFF
                        1054
E51E B84000         R   1055            MOV     AX,DATA                 ;POINT DS REG TO DATA SEG
E521 8ED8               1056            MOV     DS,AX
E523 B040               1057            MOV     AL,040H                 ;SET TIMER 2 SPK OUT, AND CASST
E525 E661               1058            OUT     PORT_B,AL               ;OUT BITS ON, CASSETTE MOT OFF
                        1059
                        1060    ;       WRITE A BIT
                        1061
E527 B0FF               1062            MOV     AL,0FFH                 ;DISABLE TIMER INTERRUPTS
E529 E621               1063            OUT     INTA01,AL
E52B B0B6               1064            MOV     AL,0B6H                 ;SEL TIM 2, LSB, MSB, MD 3
E52D E643               1065            OUT     TIMER+3,AL              ;WRITE 8253 CMD/MODE REG
E52F B80304             1066            MOV     AX,1235                 ;SET TIMER 2 CNT FOR 1000 USEC
E532 E642               1067            OUT     TIMER+2,AL              ;WRITE TIMER 2 COUNTER REG
E534 8AC4               1068            MOV     AL,AH                   ;WRITE MSB
E536 E642               1069            OUT     TIMER+2,AL
                        1070
                        1071    ;       READ CASSETTE INPUT
                        1072
E538 E462               1073            IN      AL,PORT_C               ;READ VALUE OF CASS IN BIT
E53A 2410               1074            AND     AL,10H                  ;ISOLATE FROM OTHER BITS
E53C A26800         R   1075            MOV     LAST_VAL,AL
E53F EB3E14             1076            CALL    READ_HALF_BIT
E542 E83B14             1077            CALL    READ_HALF_BIT
E545 E30C               1078            JCXZ    F8                      ; CAS_ERR
E547 81FB4005           1079            CMP     BX,MAX_PERIOD
E54B 7306               1080            JNC     F8                      ; CAS_ERR
E54D 81FB1004           1081            CMP     BX,MIN_PERIOD
E551 7309               1082            JNC     F9                      ;GO TO NEXT TEST IF OK
E553                    1083    F8:                                     ; CAS_ERR:
E553 BEABE4         R   1084            MOV     SI,OFFSET F2            ;CASSETTE WRAP FAILED
E556 B90300             1085            MOV     CX,F2L
E559 E86E01             1086            CALL    P_MSG                   ;GO PRINT ERROR MSG
                        1087    ;--------------------------------------------
                        1088    ;TEST.14
                        1089    ;       DISKETTE ATTACHMENT TEST
                        1090    ;DESCRIPTION
                        1091    ;       CHECK IF IPL DISKETTE DRIVE IS ATTACHED TO SYSTEM.  IF ATTACHED,
                        1092    ;       VERIFY STATUS OF NEC FDC AFTER A PESET.  ISSUE A RECAL AND SEEK
                        1093    ;       CMD TO FDC AND CHECK STATUS. COMPLETE SYSTEM INITIALIZATION THEN
                        1094    ;       PASS CONTROL TO THE BOOT LOADER PROGRAM.
                        1095    ;--------------------------------------------
E55C                    1096    F9:
E55C B0FC               1097            MOV     AL,0FCH                 ;ENABLE TIMER AND KBD INTS
E55E E621               1098            OUT     INTA01,AL
E560 A01000         R   1099            MOV     AL,BYTE PTR EQUIP_FLAG  ;GET SENSE SWS INFO
E563 A801               1100            TEST    AL,01H                  ;IPL DISKETTE DRIVE ATTCH?
E565 7503               1101            JNZ     F10                     ;YES - TEST DISKETTE CONTR
E567 E9B900             1102            JMP     F22                     ;NO - SKIP THIS TEST
E56A                    1103    F10:                                    ; DISK_TEST:
E56A B0BC               1104            MOV     AL,0BCH                 ;ENABLE DISKETTE, KEYBOARD,
E56C E621               1105            OUT     INTA01,AL               ; AND TIMER INTERRUPTS
E56E B400               1106            MOV     AH,0                    ;RESET NEC FDC
E570 CD13               1107            INT     13H                     ;VERIFY STATUS AFTER RESET
E572 F6C4FF             1108            TEST    AH,0FFH                 ;STATUS OK?
E575 7520               1109            JNZ     F13                     ;NO - FDC FAILED
                        1110
                        1111    ;       TURN DRIVE 0 MOTOR ON
                        1112
E577 BAF203             1113            MOV     DX,03F2H                ;GET ADDR OF FDC CARD
E57A B01C               1114            MOV     AL,1CH                  ;TURN MOTOR ON, EN DMA/INT
E57C EE                 1115            OUT     DX,AL                   ;WRITE FDC CONTROL REG
E57D 2BC9               1116            SUB     CX,CX
E57F                    1117    F11:                                    ; MOTOR_WAIT:
E57F E2FE               1118            LOOP    F11                     ;WAIT FOR 1 SECOND
E581                    1119    F12:                                    ; MOTOR_WAIT1:
E581 E2FE               1120            LOOP    F12
E583 3302               1121            XOR     DX,DX                   ;SELECT DRIVE 0
```



```
LOC OBJ                 LINE    SOURCE

E585 B501               1122            MOV     CH,1                    ;SELECT TRACK 1
E587 88163E00       R   1123            MOV     SEEK_STATUS,DL
E58B E8F30B             1124            CALL    SEEK                    ;RECALIBRATE DISKETTE
E58E 7207               1125            JC      F13                     ;GO TO ERR SUBROUTINE IF ERR
E590 B522               1126            MOV     CH,34                   ;SELECT TRACK 34
E592 E8EC08             1127            CALL    SEEK                    ;SEEK TO TRACK 34
E595 7309               1128            JNC     F14                     ;OK, TURN MOTOR OFF
E597                    1129    F13:                                    ; DSK_ERR:
E597 BEAEE4         R   1130            MOV     SI,OFFSET F3            ;GET ADDR OF MSG
E59A B90300             1131            MOV     CX,F3L                  ;GET MSG BTTE COUNT
E59D E82A01             1132            CALL    P_MSG                   ;GO PRINT ERROR MSG
                        1133
                        1134    ;       TURN DRIVE 0 MOTOR OFF
                        1135
E5A0                    1136    F14:                                    ; DRO_OFF:
E5A0 B00C               1137            MOV     AL,0CH                  ;TURN DRIVE 0 MOTOR OFF
E5A2 BAF203             1138            MOV     DX,03F2H                ; FDC CTL ADDRESS
E5A5 EE                 1139            OUT     DX,AL
                        1140
                        1141    ;       SETUP PRINTER AND RS232 BASE ADDRESSES IF DEVICE ATTACHED
                        1142
E5A6                    1143    F15:                                    ; JMP_BOOT:
E5A6 C7061A001E00   R   1144            MOV     BUFFER_HEAD,OFFSET KB_BUFFER    ;SETUP KEYBOARD PARAMETERS
E5AC C7061C001E00   R   1145            MOV     BUFFER_TAIL,OFFSET KB_BUFFER
E5B2 8DB1E4         R   1146            MOV     BP,OFFSET F4            ; PRT_SRC_TBL
E5B5 BE0000             1147            MOV     SI,0
E5B8                    1148    F16:                                    ; PRT_BASE:
E5B8 2E8B5600           1149            MOV     DX,CS:[BP]              ;GET PRINTER BASE ADDR
E5BC B0AA               1150            MOV     AL,0AAH,                ;WRITE DATA TO PORT A
E5BE EE                 1151            OUT     DX,AL
E5BF 2AC0               1152            SUB     AL,AL
E5C1 EC                 1153            IN      AL,DX                   ;READ PORT A
E5C2 3CAA               1154            CMP     AL,0AAH                 ;DATA PATTERN SAME
E5C4 7506               1155            JNE     F17                     ;NO - CHECK NEXT PRT CD
E5C6 89940800       R   1156            MOV     PRINTER_BASE[SI],DX     ;YES - STORE PRT BASE ADDR
E5CA 46                 1157            INC     SI                      ;INCREMENT TO NEXT WORD
E5CB 46                 1158            INC     SI
E5CC                    1159    F17:                                    ; NO_STORE:
E5CC 45                 1160            INC     BP                      ;POINT TO NEXT BASE ADDR
E5CD 45                 1161            INC     BP
E5CE 81FDB7E4       R   1162            CMP     BP,OFFSET F4E           ;ALL POSSIBLE ADDRS CHECKED?
E5D2 75E4               1163            JNE     F16                     ;PRT_BASE
E5D4 BB0000             1164            MOV     BX,0                    ;POINTER TO RS232 TABLE
E5D7 BAFA03             1165            MOV     DX,3FAH                 ;CHECK IF RS232 CD 1 ATTCH?
E5DA EC                 1166            IN      AL,DX                   ;READ INTR ID REG
E5DB A8F8               1167            TEST    AL,0F8H
E5DD 7508               1168            JNZ     F18
E5DF C7870000F803   R   1169            MOV     RS232_BASE[BX],3F8H     ;SETUP RS232 CD #1 ADDR
E5E5 43                 1170            INC     BX
E5E6 43                 1171            INC     BX
E5E7 BAFA02             1172    F18:    MOV     DX,2FAH                 ;CHECK IF RS232 CD 2 ATTCH
E5EA EC                 1173            IN      AL,DX                   ;READ INTERRUPT ID REG
E5EB A8F8               1174            TEST    AL,0F8H
E5ED 7508               1175            JNZ     F19                     ;BASE_END
E5EF C7870000F802   R   1176            MOV     RS232_BASE[BX],2F8H     ;SETUP RS232 CD #2
E5F5 43                 1177            INC     BX
E5F6 43                 1178            INC     BX
                        1179
                        1180    ;------ SET UP EQUIP FLAG TO INDICATE NUMBER OF PRINTERS AND RS232 CARDS
                        1181
E5F7                    1182    F19:                            ; BASE_END:
E5F7 8BC6               1183            MOV     AX,SI           ; SI HAS 2* NUMBER OF RS232
E5F9 B103               1184            MOV     CL,3            ; SHIFT COUNT
E5FB D2C8               1185            ROR     AL,CL           ; ROTATE RIGHT 3 POSITIONS
E5FD 0AC3               1186            OR      AL,BL           ; OR IN THE PRINTER COUNT
E5FF A21100         R   1187            MOV     BYTE PTR EQUIP_FLAG+1,AL        ; STORE AS SECOND BTTE
E602 BA0102             1188            MOV     DX,201H
E605 EC                 1189            IN      AL,DX
E606 A80F               1190            TEST    AL,0FH
E608 7505               1191            JNZ     F20                     ; NO_GAME_CARD
E60A 800E110010     R   1192            OR      BYTE PTR EQUIP_FLAG+1,16
E60F                    1193    F20:                                    ; NO_GAME_CARD:
                        1194
                        1195    ;       ENABLE NMI INTERRUPTS
                        1196
E60F B080               1197            MOV     AL,80H                  ;ENABLE NMI INTERRUPTS
E611 E6A0               1198            OUT     0A0H,AL
```

```
LOC OBJ                 LINE    SOURCE

E613 803E120001    R    1199            CMP     MFG_TST,1               ;MFG MODE?
E618 7406               1200            JE      F21                     ; LOAD_BOOT_STRAP
E61A BA0100             1201            MOV     DX,1                    ;
E61D E81000             1202            CALL    ERR_BEEP                ;BEEP 1 SHORT TONE
E620                    1203    F21:                                    ; LOAD_BOOT_STRAP:
E620 E9CF00             1204            JMP     BOOT_STRAP              ;GO TO THE BOOT LOADER
E623                    1205    F22:                                    ; LOOP_POD:
E623 803E120001    R    1206            CMP     MFG_TST,1               ;MANUFACTURING TEST MODE?
E628 7503               1207            JNE     F23                     ;NO - GO TO BOOT LOADER
E62A E9EEFA             1208            JMP     START                   ;TES - LOOP POWER-ON-DIAGS
E62D                    1209    F23:                                    ; GO_TO_BOOT:
E620 E976FF             1210            JMP     F1S                     ; JMP_BOOT
                        1211    ;--------------------------------------
                        1212    ; INITIAL RELIABILITY TEST -- SUBROUTINES
                        1213    ;--------------------------------------
                        1214            ASSUME  CS:CODE,DS:DATA
                        1215    ;--------------------------------------
                        1216    ;       SUBROUTINES FOR POWER ON DIAGNOSTICS
                        1217    ;--------------------------------------
                        1218    ;       THIS PROCEDURE WILL ISSUE ONE LONG TONE (3 SECS) AND ONE OR
                        1219    ;       MORE SHORT TONES (1 SEC) TO INDICATE A FAILURE ON THE PLANAR
                        1220    ;       BOARD, A BAD RAM MODULE, OR A PROBLEM WITH THE CRT.
                        1221    ;ENTRY PARAMETERS:
                        1222    ;       DH = NUMBER OF LONG TONES TO BEEP
                        1223    ;       DL = NUMBER OF SHORT TONES TO BEEP.
                        1224    ;--------------------------------------
E630                    1225    ERR_BEEP  PROC  NEAR
E630 9C                 1226            PUSHF                           ;SAVE FLAGS
E631 FA                 1227            CLI                             ;DISABLE SYSTEM INTERRUPTS
E632 1E                 1228            PUSH    DS                      ;SAVE DS REG CONTENTS
E633 B84000        R    1229            MOV     AX,DATA                 ;POINT DS TO DATA SEG
E636 8ED8               1230            MOV     DS,AX
E638 0AF6               1231            OR      DH,DH           ; ANY LONG ONES TO BEEP
E63A 7418               1232            JZ      G3              ; NO, DO THE SHORT ONES
E63C                    1233    G1:                             ; LONG_BEEP:
E63C B306               1234            MOV     BL,6            ; COUNTER FOR BEEPS
E63E E82500             1235            CALL    BEEP            ; DO THE BEEP
E641 E2FE               1236    G2:     LOOP    G2              ; DELAY BETWEEN BEEPS
E643 FECE               1237            DEC     DH              ; ANY MORE TO DO
E645 75F5               1238            JNZ     G1              ; DO IT
E647 803E120001    R    1239            CMP     MFG_TST,1       ; MFG TEST MODE?
E64C 7506               1240            JNE     G3              ; YES - CONTINUE BEEPING SPEAKER
E64E B0CD               1241            MOV     AL,0CDH         ; STOP BLINKING LED
E650 E661               1242            OUT     PORT_B,AL
E652 EBEB               1243            JMP     SHORT G1
E654                    1244    G3:                             ; SHORT_BEEP:
E654 B301               1245            MOV     BL,1            ; COUNTER FOR A SHORT BEEP
E656 E80D00             1246            CALL    BEEP            ; DO THE SOUND
E659 E2FE               1247    G4:     LOOP    G4              ; DELAT BETWEEN BEEPS
E65B FECA               1248            DEC     DL              ; DONE WITH SHORTS
E650 75F5               1249            JNZ     G3              ; DO SOME MORE
E65F E2FE               1250    G5:     LOOP    G5              ; LONG DELAY BEFORE RETURN
E661 E2FE               1251    G6:     LOOP    G6
E663 1F                 1252            POP     DS              ;RESTORE ORIG CONTENTS OF DS
E664 9D                 1253            POPF                    ;RESTORE FLAGS TO ORIG SETTINGS
E665 C3                 1254            RET                     ; RETURN TO CALLER
                        1255    ERR_BEEP        ENDP
                        1256
                        1257    ;       ROUTINE TO SOUND BEEPER
                        1258
E666                    1259    BEEP    PROC    NEAR
E666 B086               1260            MOV     AL,10110110B            ;SEL TIM 2,LSB,MSB,BINART
E668 E643               1261            OUT     TIMER+3,AL              ;WRITE THE TIMER MODE REG
E66A BB3305             1262            MOV     AX,533H                 ;DIVISOR FOR 1000 HZ
E66D E642               1263            OUT     TIMER+2,AL              ;WRITE TIMER 2 CNT - LSB
E66F 8AC4               1264            MOV     AL,AH
E671 E642               1265            OUT     TIMER+2,AL              ;WRITE TIMER 2 CNT - MSB
E673 E461               1266            IN      AL,PORT_B               ;GET CURRENT SETTING OF PORT
E675 8AE0               1267            MOV     AH,AL           ; SAVE THAT SETTINGH
E677 0C03               1268            OR      AL,03           ;TURN SPEAKER ON
E679 E661               1269            OUT     PORT_B,AL
E678 2BC9               1270            SUB     CX,CX           ;SET CNT TO WAIT 500 MS
E670 E2FE               1271    G7:     LOOP    G7              ;DELAT BEFORE TURNING OFF
E67F FECB               1272            DEC     BL              ;DELAY CNT EXPIRED?
E681 75FA               1273            JNZ     G7              ;NO - CONTINUE BEEPING SPK
E683 8AC4               1274            MOV     AL,AH           ; RECOVER VALUE OF PORT
```



```
LOC OBJ                 LINE    SOURCE


E685 E661               1275            OUT     PORT_B,AL
E687 C3                 1276            RET                             ;RETURN TO CALLER
                        1277    BEEP    ENDP
                        1278    ;--------------------------------------
                        1279    ;       THIS PROCEDURE WILL SEND A SOFTWARE RESET TO THE KEYBOARD.
                        1280    ;       SCAN CODE  AA' SHOULD BE RETURNED TO THE CPU.
                        1281    ;--------------------------------------
E688                    1282    KBD_RESET       PROC    NEAR
E688 B00C               1283            MOV     AL,0CH                  ;SET KBD CLK LINE LOW
E68A E661               1284            OUT     PORT_B,AL               ;WRITE 8255 PORT B
E68C B95629             1285            MOV     CX,10582                ;HOLD KBD CLK LOW FOR 20 MS
E68F E2FE               1286    G8:     LOOP    G8                      ;LOOP FOR 20 MS
E691 B0CC               1287            MOV     AL,0CCH                 ;SET CLK, ENABLE LINES HIGH
E693 E661               1288            OUT     PORT_B,AL
E695                    1289    SP_TEST:                ; ENTRY FOR MANUFACTURING TEST 2
E695 B04C               1290            MOV     AL,4CH                  ;SET KBD CLK HIGH, ENABLE LOW
E697 E661               1291            OUT     PORT_B,AL
E699 B0FD               1292            MOV     AL,0FDH                 ;ENABLE KETBOARD INTERRUPTS
E69B E621               1293            OUT     INTA01,AL               ;WRITE 8259 IMR
E69D FB                 1294            STI                             ;ENABLE SYSTEM INTERRUPTS
E69E B400               1295            MOV     AH,0                    ;RESET INTERRUPT INDICATOR
E6A0 2BC9               1296            SUB     CX,CX                   ;SETUP INTERRUPT TIMEOUT CNT
E6A2 F6C4FF             1297    G9:     TEST    AH,0FFH                 ;DID A KEYBOARD INTR OCCUR?
E6A5 7502               1298            JNZ     G10                     ;YES - READ SCAN CODE RETURNED
E6A7 E2F9               1299            LOOP    G9                      ;NO - LOOP TILL TIMEOUT
E6A9 E460               1300    G10:    IN      AL,PORT_A               ;READ KETBOARD SCAN CODE
E6AB 8AD8               1301            MOV     BL,AL                   ;SAVE SCAN CODE JUST READ
E6AD B0CC               1302            MOV     AL,0CCH                 ;CLEAR KETBOARD
E6AF E661               1303            OUT     PORT_B,AL
E6B1 C3                 1304            RET                             ;RETURN TO CALLER
                        1305    KBD_RESET       ENDP
                        1306    ;--------------------------------------
                        1307    ;       BLINK LED PROCEDURE FOR MFG BURN-IN AND RUN-IN TESTS
                        1308    ;       (LED WILL BLINK APPROXIMATELY .25 SECOND)
                        1309    ;--------------------------------------
E6B2                    1310    BLINK_INT       PROC    NEAR
E6B2 FB                 1311            STI
E6B3 51                 1312            PUSH    CX                      ;SAVE CX REG CONTENTS
E6B4 50                 1313            PUSH    AX                      ;SAVE AX REG CONTENTS
E6B5 E461               1314            IN      AL,PORT_B               ;READ CURRENT VAL OF PORT B
E6B7 24BF               1315            AND     AL,0BFH
E6B9 E661               1316            OUT     PORT_B,AL               ;BLINK LED
E6BB 2BC9               1317            SUB     CX,CX
E6BD E2FE               1318    G11:    LOOP    G11
E6BF 0C40               1319            OR      AL,40H                  ;STOP BLINKING LED
E6C1 E661               1320            OUT     PORT_B,AL
E6C3 B020               1321            MOV     AL,EOI
E6C5 E620               1322            OUT     INTA00,AL
E6C7 58                 1323            POP     AX                      ;RESTORE AX REG
E6C8 59                 1324            POP     CX                      ;RESTORE CX REG
E6C9 CF                 1325            IRET
                        1326    BLINK_INT       ENDP
                        1327    ;--------------------------------------
                        1328    ;       THIS SUBROUTINE WILL PRINT A MESSAGE ON THE DISPLAT
                        1329    ;
                        1330    ;ENTRY REQUIREMENTS:
                        1331    ;       SI = OFFSET(ADDRESS) OF MESSAGE BUFFER
                        1332    ;       CX = MESSAGE BYTE COUNT
                        1333    ;       MAXIMUM MESSAGE LENGTH IS 36 CHARACTERS
                        1334    ;--------------------------------------
E6CA                    1335    P_MSG   PROC    NEAR
E6CA B84000        R    1336            MOV     AX,DATA                 ;POINT DS TO DATA SEG
E6CD 8ED8               1337            MOV     DS,AX
E6CF 803E120001    R    1338            CMP     MFG_TST,1               ;MFG TEST MODE?
E6D4 7505               1339            JNE     G12                     ;NO - DISPLAT ERROR MSG
E6D6 B601               1340            MOV     DH,1                    ;YES - SETUP TO BEEP SPEAKER
E6D8 E955FF             1341            JMP     ERR_BEEP                ;YES - BEEP SPEAKER
E6DB                    1342    G12:                            ; WRITE_MSG:
E6DB 2EBA04             1343            MOV     AL,CS:[SI]              ;PUT CHAR IN AL
E6DE 46                 1344            INC     SI                      ;POINT TO NEXT CHAR
E6DF B700               1345            MOV     BH,0                    ;SET PAGE # TO ZERO
E6E1 B40E               1346            MOV     AH,14                   ;WRITE CHAR (TTY-INTERFACE)
E6E3 CD10               1347            INT     10H                     ;CALL VIDEO_IO
E6E5 E2F4               1348            LOOP    G12                     ;CONTINUE TILL MSG WRITTEN
E6E7 B80D0E             1349            MOV     AX,0E0DH                ;POSITION CURSOR TO NEXT LINE
E6EA CD10               1350            INT     10H                     ;SEND CARRIAGE RETURN AND
```

```
LOC OBJ                 LINE    SOURCE

E6EC B80A0E             1351            MOV     AX,0E0AH                ;LINE FEED CHARS
E6EF CD10               1352            INT     10H
E6F1 C3                 1353            RET
                        1354    P_MSG   ENDP
                        1355    ;--- INT 19 ------------------------------
                        1356    ;BOOT STRAP LOADER
                        1357    ;       IF A 5 1/4" DISKETTE DRIVE IS AVAILABLE
                        1358    ;       ON THE SYSTEM, TRACK 0, SECTOR 1 IS READ INTO THE
                        1359    ;       BOOT LOCATION (SEGMENT 0, OFFSET 7C00)
                        1360    ;       AND CONTROL IS TRANSFERRED THERE.
                        1361    ;
                        1362    ;       IF THERE IS NO DISKETTE DRIVE, OR IF THERE IS
                        1363    ;       IS A HARDWARE ERROR CONTROL IS TRANSFERRED
                        1364    ;       TO THE CASSETTE BASIC ENTRY POINT.
                        1365    ;
                        1366    ; IPL ASSUMPTIONS
                        1367    ;       8255 PORT 60H BIT 0
                        1368    ;       = 1 IF IPL FROM DISKETTE
                        1369    ;----------------------------------------
                        1370            ASSUME  CS:CODE,DS:DATA
E6F2                    1371    BOOT_STRAP      PROC    NEAR
                        1372
E6F2 FB                 1373            STI                     ; ENABLE INTERRUPTS
E6F3 B84000         R   1374            MOV     AX,DATA         ; ESTABLISH ADDRESSING
E6F6 8ED8               1375            MOV     DS,AX
E6F8 A11000         R   1376            MOV     AX,EQUIP_FLAG   ; GET THE EQUIPMENT SWITCHES
E6FB A801               1377            TEST    AL,1            ; ISOLATE IPL SENSE SWITCH
E6FD 7423               1378            JZ      H3              ; GO TO CASSETTE BASIC ENTRY POINT
                        1379
                        1380    ;------ MUST LOAD SYSTEM FROM DISKETTE -- CX HAS RETRY COUNT
                        1381
E6FF B90400             1382            MOV     CX,4            ; SET RETRY COUNT
E702                    1383    H1:                             ; IPL_SYSTEM
E702 51                 1384            PUSH    CX              ; SAVE RETRY COUNT
E703 B400               1385            MOV     AH,0            ; RESET THE DISKETTE SYSTEM
E705 CD13               1386            INT     13H             ; DISKETTE_IO
E707 7214               1387            JC      H2              ; IF ERROR, TRY AGAIN
E709 B402               1388            MOV     AH,2            ; READ IN THE SINGLE SECTOR
E70B BB0000             1389            MOV     BX,0            ; TO THE BOOT LOCATION
E70E 8EC3               1390            MOV     ES,BX
E710 BB007C             1391            MOV     BX,OFFSET BOOT_LOCN
E713 BA0000             1392            MOV     DX,0            ; DRIVE 0, HEAD 0
E716 B90100             1393            MOV     CX,1            ; SECTOR 1, TRACK 0
E719 B001               1394            MOV     AL,1            ; READ ONE SECTOR
E71B CD13               1395            INT     13H             ; DISKETTE_IO
E71D 59                 1396    H2:     POP     CX              ; RECOVER RETRY COUNT
E71E 7304               1397            JNC     H4              ; CF SET BY UNSUCCESSFUL READ
E720 E2E0               1398            LOOP    H1              ; DO IT FOR RETRY TIMES
                        1399
                        1400    ;------ UNABLE TO IPL FROM THE DISKETTE
                        1401
E722                    1402    H3:                             ; CASSETTE_JUMP:
E722 CD18               1403            INT     18H             ; USE INTERRUPT VECTOR TO GET TO BASIC
                        1404
                        1405    ;------ IPL WAS SUCCESSFUL
                        1406
E724                    1407    H4:
E724 EA007C0000         1408            JMP     BOOT_LOCN
                        1409    BOOT_STRAP      ENDP
                        1410    ;-----INT 14----------------------------------
                        1411    ;RS232_IO
                        1412    ;       THIS ROUTINE PROVIDES BYTE STREAM I/O TO THE COMMUNICATIONS
                        1413    ;       PORT ACCORDING TO THE PARAMETERS:
                        1414    ;       (AH)=0  INITIALIZE THE COMMUNICATIONS PORT
                        1415    ;               (AL) HAS PARMS FOR INITIALIZATION
                        1416    ;
                        1417    ;               7       6       5       4       3       2       1       0
                        1418    ;               ----- BAUD RATE --      --PARITY--     STOPBIT   --WORD LENGTH--
                        1419
                        1420    ;               000 - 110               X0 - NONE       0 - 1   10 - 7 BITS
                        1421    ;               001 - 150               01 - ODD        1 - 2   11 - 8 BITS
                        1422    ;               010 - 300               11 - EVEN
                        1423    ;               011 - 600
                        1424    ;               100 - 1200
                        1425    ;               101 - 2400
                        1426    ;               110 - 4800
                        1427    ;               111 - 9600
```



```
LOC OBJ                 LINE    SOURCE

                        1428    ;               ON RETURN, CONDITIONS SET AS IN CALL TO COMMO STATUS (AH=3)
                        1429    ;       (AH)=1  SEND THE CHARACTER IN (AL) OVER THE COMMO LINE
                        1430    ;               (AL) REGISTER IS PRESERVED
                        1431    ;               ON EXIT, BIT 7 OF AH IS SET IF THE ROUTINE WAS UNABLE TO
                        1432    ;                       TO TRANSMIT THE BYTE OF DATA OVER THE LINE.  THE
                        1433    ;                       REMAINDER OF AH IS SET AS IN A STATUS REQUEST,
                        1434    ;                       REFELECTING THE CURRENT STATUS OF THE LINE.
                        1435    ;       (AH)=2  RECEIVE A CHARACTER IN (AL) FROM COMMO LINE BEFORE
                        1436    ;                       RETURNING TO CALLER
                        1437    ;               ON EXIT, AH HAS THE CURRENT LINE STATUS, AS SET BY THE
                        1438    ;                       THE STATUS ROUTINE, EXCEPT THAT THE ONLY BITS
                        1439    ;                       LEFT ON ARE THE ERROR BITS (7,4,3,2,1)
                        1440    ;                       IN THIS CASE, THE TIME OUT BIT INDICATES DATA SET
                        1441    ;                       READY WAS NOT RECEIVED.
                        1442    ;                       THUS, AH IS NON ZERO ONLY WHEN AN ERROR OCCURRED.
                        1443    ;       (AH)=3  RETURN THE COMMO PORT STATUS IN (AX)
                        1444    ;               AH CONTAINS THE LINE CONTROL STATUS
                        1445    ;               BIT 7 = TIME OUT
                        1446    ;               BIT 6 = TRANS SHIFT REGISTER EMPTY
                        1447    ;               BIT 5 = TRAN HOLDING REGISTER EMPTY
                        1448    ;               BIT 4 = BREAK DETECT
                        1449    ;               BIT 3 = FRAMING ERROR
                        1450    ;               BIT 2 = PARITY ERROR
                        1451    ;               BIT 1 = OVERRUN ERROR
                        1452    ;               BIT 0 = DATA READY
                        1453    ;               AL CONTAINS THE MODEM STATUS
                        1454    ;               BIT 7 = RECEIVED LINE SIGNAL DETECT
                        1455    ;               BIT 6 = RING INDICATOR
                        1456    ;               BIT 5 = DATA SET READY
                        1457    ;               BIT 4 = CLEAR TO SEND
                        1458    ;               BIT 3 = DELTA RECEIVE LINE SIGNAL DETECT
                        1459    ;               BIT 2 = TRAILING EDGE RING DETECTOR
                        1460    ;               BIT 1 = DELTA DATA SET READY
                        1461    ;               BIT 0 = DELTA CLEAR TO SEND
                        1462
                        1463    ;       (DX) = PARAMETER INDICATING WHICH RS232 CARD (0,1 ALLOWED)
                        1464    ; DATA AREA RS232_BASE CONTAINS THE BASE ADDRESS OF THE 8250 ON THE CARD
                        1465    ;       LOCATION 400H CONTAINS UP TO 4 RS232 ADDRESSES POSSIBLE
                        1466    ;OUTPUT
                        1467    ;               AX      MODIFIED ACCORDING TO PARMS OF CALL
                        1468    ;               ALL OTHERS UNCHANGED
                        1469    ;-----------------------------------------------
                        1470            ASSUME  CS:CODE,DS:DATA
E729                    1471    A1      LABEL   WORD
E729 1704               1472                    DW      1047    ; 110 BAUD      ; TABLE OF INIT VALUE
E72B 0003               1473                    DW      768     ; 150
E72D 8001               1474                    DW      384     ; 300
E72F C000               1475                    DW      192     ; 600
E731 6000               1476                    DW      96      ; 1200
E733 3000               1477                    DW      48      ; 2400
E735 1800               1478                    DW      24      ; 4800
E737 0C00               1479                    DW      12      ; 9600
                        1480
E739                    1481    RS232_IO        PROC    FAR
                        1482
                        1483    ;------ VECTOR TO APPROPRIATE ROUTINE
                        1484
E739 FB                 1485            STI                     ; INTERRUPTS BACK ON
E73A 1E                 1486            PUSH    DS              ; SAVE SEGMENT
E73B 52                 1487            PUSH    DX
E73C 56                 1488            PUSH    SI
E73D 57                 1489            PUSH    DI
E73E 51                 1490            PUSH    CX
E73F 8BF2               1491            MOV     SI,DX           ; RS232 VALUE TO SI
E741 01E6               1492            SHL     SI,1            ; WORD OFFSET
E743 BA4000         R   1493            MOV     DX,DATA
E746 8ED A              1494            MOV     DS,DX           ; SET UP OUR SEGMENT
E748 8B940000       R   1495            MOV     DX,RS232_BASE[SI]       ; GET BASE ADDRESS
E74C 0BD2               1496            OR      DX,DX           ; TEST FOR 0 BASE ADDRESS
E74E 7416               1497            JZ      A3              ; RETURN
E750 0AE4               1498            OR      AH,AH           ; TEST FOR (AH)=0
E752 7418               1499            JZ      A4              ; COMMUN INIT
E754 FECC               1500            DEC     AH              ; TEST FOR (AH)=1
E756 744E               1501            JZ      A5              ; SEND AL
E758 FECC               1502            DEC     AH              ; TEST FOR (AH)=2
E75A 7503               1503            JNZ     A2
E75C E98900             1504            JMP     A12             ; RECEIVE INTO AL
```

```
LOC OBJ                 LINE    SOURCE

E75F                    1505    A2:
E75F FECC               1506            DEC     AH              ; TEST FOR (AH)=3
E761 7503               1507            JNZ     A3
E763 E9B900             1508            JMP     A18             ; COMMUNICATION STATUS
E766                    1509    A3:                             ; RETURN FROM RS232
E766 59                 1510            POP     CX
E767 5F                 1511            POP     DI
E768 5E                 1512            POP     SI
E769 5A                 1513            POP     DX
E76A 1F                 1514            POP     DS
E76B CF                 1515            IRET                    ; RETURN TO CALLER, NO ACTION
                        1516
                        1517    ;------ INITIALIZE THE COMMUNICATIONS PORT
                        1518
E76C                    1519    A4:
E76C 8AE0               1520            MOV     AH,AL           ; SAVE INIT PARMS IN AH
E76E 83C203             1521            ADD     DX,3            ; POINT TO 8250 CONTROL REGISTER
E771 B080               1522            MOV     AL,80H
E773 EE                 1523            OUT     DX,AL           ; SET DLAB=1
                        1524
                        1525    ;------ DETERMINE BAUD RATE DIVISOR
                        1526
E774 8AD4               1527            MOV     DL,AH           ; GET PARMS TO DL
E776 D0C2               1528            ROL     DL,1
E778 D0C2               1529            ROL     DL,1            ; GET BAUD RATE TERM TO LOW BITS
E77A D0C2               1530            ROL     DL,1
E77C D0C2               1531            ROL     DL,1            ; *2 FOR WORD TABLE ACCESS
E77E 81E20E00           1532            AND     DX,0EH          ; ISOLATE THEM
E782 BF29E7     R       1533            MOV     DI,OFFSET A1    ; BASE OF TABLE
E785 03FA               1534            ADD     DI,DX           ; PUT INTO INDEX REGISTER
E787 8B940000   R       1535            MOV     DX,RS232_BASE[SI]       ; POINT TO HIGH ORDER OF DIVISOR
E78B 42                 1536            INC     DX
E78C 2E8A4501           1537            MOV     AL,CS:[DI]+1    ; GET HIGH ORDER OF DIVISOR
E790 EE                 1538            OUT     DX,AL           ; SET MS OF DIV TO 0
E791 4A                 1539            DEC     DX
E792 2E8A05             1540            MOV     AL,CS:[DI]      ; GET LOW ORDER OF DIVISOR
E795 EE                 1541            OUT     DX,AL           ; SET LOW OF DIVISOR
E796 83C203             1542            ADD     DX,3
E799 8AC4               1543            MOV     AL,AH           ; GET PARMS BACK
E79B 241F               1544            AND     AL,01FH         ; STRIP OFF THE BAUD BITS
E79D EE                 1545            OUT     DX,AL           ; LINE CONTROL TO 8 BITS
E79E 83EA02             1546            SUB     DX,2
E7A1 B000               1547            MOV     AL,0
E7A3 EE                 1548            OUT     DX,AL           ; INTERRUPT ENABLES ALL OFF
E7A4 EB79               1549            JMP     SHORT A18       ; COM_STATUS
                        1550
                        1551    ;------ SEND CHARACTER IN (AL) OVER COMMO LINE
                        1552
E7A6                    1553    A5:
E7A6 50                 1554            PUSH    AX              ; SAVE CHAR TO SEND
E7A7 83C204             1555            ADD     DX,4            ; MODEM CONTROL REGISTER
E7AA B003               1556            MOV     AL,3            ; DTR AND RTS
E7AC EE                 1557            OUT     DX,AL           ; DATA TERMINAL READY, REQUEST TO SEND
E7AD 33C9               1558            XOR     CX,CX           ; INITIALIZE TIME OUT COUNT
E7AF 83C202             1559            ADD     DX,2            ; MODEM STATUS REGISTER
E7B2                    1560    A6:                             ; WAIT_DATA_SET_READY
E7B2 EC                 1561            IN      AL,DX           ; GET MODEM STATUS
E7B3 A820               1562            TEST    AL,20H          ; DATA SET READY
E7B5 7508               1563            JNZ     A7              ; TEST_CLEAR_TO_SEND
E7B7 E2F9               1564            LOOP    A6              ; WAIT_DATA_SET_READY
E7B9 58                 1565            POP     AX
E7BA 80CC50             1566            OR      AH,80           ; INDICATE TIME OUT
E7BD EBA7               1567            JMP     A3              ; RETURN
E7BF                    1568    A7:                             ; TEST_CLEAR_TO_SEND
E7BF 2BC9               1569            SUB     CX,CX
E7C1                    1570    A8:                             ; WAIT_CLEAR_TO_SEND
E7C1 EC                 1571            IN      AL,DX           ; GET MODEM STATUS
E7C2 A810               1572            TEST    AL,10H          ; TEST CLEAR TO SEND
E7C4 7508               1573            JNZ     A9              ; CLEAR_TO_SEND
E7C6 E2F9               1574            LOOP    A8              ; WAIT_CLEAR_TO_SEND
E7C8 58                 1575            POP     AX              ; TIME OUT HAS OCCURRED
E7C9 80CC80             1576            OR      AH,80H
E7CC EB98               1577            JMP     A3              ; RETURN
E7CE                    1578    A9:                             ; CLEAR_TO_SEND
E7CE 4A                 1579            DEC     DX              ; LINE STATUS REGISTER
E7CF 2BC9               1580            SUB     CX,CX           ; INITIALIZE WAIT COUNT
E7D1                    1581    A10:                            ; WAIT_SEND
```



```
LOC OBJ                 LINE    SOURCE

E7D1 EC                 1582            IN      AL,DX           ; GET STATUS
E7D2 A820               1583            TEST    AL,20H          ; IS TRANSMITTER READY
E7D4 7508               1584            JNZ     A11             ; OUT_CHAR
E7D6 E2F9               1585            LOOP    A10             ; GO BACK FOR MORE, AND TEST FOR TIME OUT
E7D8 58                 1586            POP     AX              ; RECOVER ORIGINAL INPUT
E7D9 80CC80             1587            OR      AH,80H          ; SET THE TIME OUT BIT
E7DC EB88               1588            JMP     A3              ; RETURN
E7DE                    1589    A11:                            ; OUT_CHAR
E7DE 83EA05             1590            SUB     DX,5            ; DATA PORT
E7E1 59                 1591            POP     CX              ; RECOVER IN CX TEMPORARILY
E7E2 8AC1               1592            MOV     AL,CL           ; GET OUT CHAR TO AL FOR OUT, STATUS IN AH
E7E4 EE                 1593            OUT     DX,AL           ; OUTPUT CHARACTER
E7E5 E97EFF             1594            JMP     A3              ; RETURN
                        1595
                        1596    ;------ RECEIVE CHARACTER FROM COMMO LINE
                        1597
E7E8                    1598    A12:
E7E8 802671007F   R     1599            AND     BIOS_BREAK,07FH         ; TURN OFF BREAK BIT IN BYTE
E7ED 83C204             1600            ADD     DX,4            ; MODEM CONTROL REGISTER
E7F0 B001               1601            MOV     AL,1            ; DATA TERMINAL READY
E7F2 EE                 1602            OUT     DX,AL
E7F3 83C202             1603            ADD     DX,2            ; MODEM STATUS REGISTER
E7F6 2BC9               1604            SUB     CX,CX           ; ESTABLISH TIME OUT COUNT
E7F8                    1605    A13:                            ; WAIT_DSR
E7F8 EC                 1606            IN      AL,DX           ; MODEM STATUS
E7F9 A820               1607            TEST    AL,20H          ; DATA SET READY
E7FB 7507               1608            JNZ     A15             ; IS IT READY YET
E7FD E2F9               1609            LOOP    A13             ; WAIT UNTIL IT IS
E7FF                    1610    A14:                            ; TIME_OUT_ERR
E7FF B480               1611            MOV     AH,80H          ; SET TIME OUT ERROR
E801 E962FF             1612            JMP     A3              ; RETURN WITH ERROR
E804                    1613    A15:                            ; WAIT_DSR_END
E804 4A                 1614            DEC     DX              ; LINE STATUS REGISTER
E805                    1615    A16:                            ; WAIT_RECV
E805 EC                 1616            IN      AL,DX           ; GET STATUS
E806 A801               1617            TEST    AL,1            ; RECEIVE BUFFER FULL
E808 7509               1618            JNZ     A17             ; GET CHAR
E80A F606710080   R     1619            TEST    BIOS_BREAK,80H  ; TEST FOR BREAK KEY
E80F 74F4               1620            JZ      A16             ; LOOP IF NOT
E811 EBEC               1621            JMP     A14             ; SET TIME OUT ERROR
E813                    1622    A17:                            ; GET_CHAR
E813 241E               1623            AND     AL,00011110B    ; TEST FOR ERROR CONDITIONS ON RECV CHAR
E815 8AE0               1624            MOV     AH,AL           ; SAVE THIS PART OF STATUS FOR LATER OPERATION
E817 8B940000   R       1625            MOV     DX,RS232_BASE[SI]       ; DATA PORT
E81B EC                 1626            IN      AL,DX           ; GET CHARACTER FROM LINE
E81C E947FF             1627            JMP     A3              ; RETURN
                        1628
                        1629    ;------ COMMO PORT STATUS ROUTINE
                        1630
E81F                    1631    A18:
E81F 8B940000   R       1632            MOV     DX,RS232_BASE[SI]
E823 83C205             1633            ADD     DX,5            ; CONTROL PORT
E826 EC                 1634            IN      AL,DX           ; GET LINE CONTROL STATUS
E827 8AE0               1635            MOV     AH,AL           ; PUT IN AH FOR RETURN
E829 42                 1636            INC     DX              ; POINT TO MODEM STATUS REGISTER
E82A EC                 1637            IN      AL,DX           ; GET MODEM CONTROL STATUS
E82B E938FF             1638            JMP     A3              ; RETURN
                        1639    RS232_IO        ENDP
                        1640    ;---- INT 16 ----------------------------------
                        1641    ; KEYBOARD I/O
                        1642    ;       THESE ROUTINES PROVIDE KEYBOARD SUPPORT
                        1643    ; INPUT
                        1644    ;       (AH)=0  READ THE NEXT ASCII CHARACTER STRUCK FROM THE KEYBOARD
                        1645    ;               RETURN THE RESULT IN (AL), SCAN CODE IN (AH)
                        1646    ;       (AH)=1  SET THE Z FLAG TO INDICATE IF AN ASCII CHARACTER IS AVAILABLE
                        1647    ;               TO BE READ.
                        1648    ;               (ZF)=1 -- NO CODE AVAILABLE
                        1649    ;               (ZF)=0 -- CODE IS AVAILABLE
                        1650    ;               IF ZF = 0, THE NEXT CHARACTER IN THE BUFFER TO BE READ IS
                        1651    ;               IN AX, AND THE ENTRY REMAINS IN THE BUFFER
                        1652    ;       (AH)=2  RETURN THE CURRENT SHIFT STATUS IN AL REGISTER
                        1653    ;               THE BIT SETTINGS FOR THIS CODE ARE INDICATED IN THE
                        1654    ;               THE EQUATES FOR KB_FLAG
                        1655    ; OUTPUT
                        1656    ;       AS NOTED ABOVE, ONLY AX AND FLAGS CHANGED
                        1657    ;       ALL REGISTERS RETAINED
                        1658    ;----------------------------------------
```

```
LOC OBJ                 LINE    SOURCE


                        1659            ASSUME  CS:CODE,DS:DATA
E82E                    1660    KEYBOARD_IO     PROC    FAR
E82E FB                 1661            STI                     ; INTERRUPTS BACK ON
E82F 1E                 1662            PUSH    DS              ; SAVE CURRENT DS
E830 53                 1663            PUSH    BX              ; SAVE BX TEMPORARILY
E831 BB4000         R   1664            MOV     BX,DATA         ;
E834 8EDB               1665            MOV     DS,BX           ; ESTABLISH POINTER TO DATA REGION
E836 0AE4               1666            OR      AH,AH           ; AH=0
E838 740B               1667            JZ      K1              ; ASCII_READ
E83A FECC               1668            DEC     AH              ; AH=1
E83C 7420               1669            JZ      K2              ; ASCII_STATUS
E83E FECC               1670            DEC     AH              ; AH=2
E840 7420               1671            JZ      K3              ; SHIFT_STATUS
E842 5B                 1672            POP     BX              ; RECOVER REGISTER
E843 1F                 1673            POP     DS
E844 CF                 1674            IRET                    ; INVALID COMMAND
                        1675
                        1676    ;------ READ THE KEY TO FIGURE OUT WHAT TO DO
                        1677
E845                    1678    K1:                             ; ASCII READ
E845 FB                 1679            STI                     ; INTERRUPTS BACK ON DURING LOOP
E846 90                 1680            NOP                     ; ALLOW AN INTERRUPT TO OCCUR
E847 FA                 1681            CLI                     ; INTERRUPTS BACK OFF
E848 8B1E1A00       R   1682            MOV     BX,BUFFER_HEAD  ; GET POINTER TO HEAD OF BUFFER
E84C 3B1E1C00       R   1683            CMP     BX,BUFFER_TAIL  ; TEST END OF BUFFER
E850 74F3               1684            JZ      K1              ; LOOP UNTIL SOMETHING IN BUFFER
E852 8B07               1685            MOV     AX,[BX]         ; GET SCAN CODE AND ASCII CODE
E854 E81E00             1686            CALL    K4              ; MOVE POINTER TO NEXT POSITION
E857 891E1A00       R   1687            MOV     BUFFER_HEAD,BX  ; STORE VALUE IN VARIABLE
E85B 5B                 1688            POP     BX              ; RECOVER REGISTER
E85C 1F                 1689            POP     DS              ; RECOVER SEGMENT
E85D CF                 1690            IRET                    ; RETURN TO CALLER
                        1691
                        1692    ;------ ASCII STATUS
                        1693
E85E                    1694    K2:
E85E FA                 1695            CLI                     ; INTERRUPTS OFF
E85F 8B1E1A00       R   1696            MOV     BX,BUFFER_HEAD  ; GET HEAD POINTER
E863 3B1E1C00       R   1697            CMP     BX,BUFFER_TAIL  ; IF EQUAL (Z=1) THEN NOTHING THERE
E867 8B07               1698            MOV     AX,[BX]
E869 FB                 1699            STI                     ; INTERRUPTS BACK ON
E86A 5B                 1700            POP     BX              ; RECOVER REGISTER
E86B 1F                 1701            POP     DS              ; RECOVER SEGMENT
E86C CA0200             1702            RET     2               ; THROW AWAY FLAGS
                        1703
                        1704    ;------ SHIFT STATUS
                        1705
E86F                    1706    K3:
E86F A01700         R   1707            MOV     AL,KB_FLAG      ; GET THE SHIFT STATUS FLAGS
E872 5B                 1708            POP     BX              ; RECOVER REGISTER
E873 1F                 1709            POP     DS              ; RECOVER REGISTERS
E874 CF                 1710            IRET                    ; RETURN TO CALLER
                        1711    KEYBOARD_IO     ENDP
                        1712
                        1713    ;------ INCREMENT A BUFFER POINTER
                        1714
E875                    1715    K4      PROC    NEAR
E875 83C302             1716            ADD     BX,2                    ; MOVE TO NEXT WORD IN LIST
E878 3B1E3E00       R   1717            CMP     BX,OFFSET KB_BUFFER_END ; AT END OF BUFFER?
E87C 7503               1718            JNE     K5                      ; NO, CONTINUE
E87E BB1E00         R   1719            MOV     BX,OFFSET KB_BUFFER     ; YES, RESET TO BUFFER BEGINNING
E881                    1720    K5:
E881 C3                 1721            RET
                        1722    K4      ENDP
                        1723
                        1724    ;------ TABLE OF SHIFT KEYS AND MASK VALUES
                        1725
E882                    1726    K6      LABEL   BYTE
E882 52                 1727            DB      INS_KEY                 ; INSERT KEY
E883 3A45463810         1728            DB      CAPS_KEY,NUM_KEY,SCROLL_KEY,ALT_KEY,CTL_KEY
E888 2A36               1729            DB      LEFT_KEY,RIGHT_KEY
  0008                  1730    K6L     EQU     $-K6
                        1731
                        1732    ;------ SHIFT_MASK_TABLE
                        1733
E88A                    1734    K7      LABEL   BYTE
E88A 80                 1735            DB      INS_SHIFT               ; INSERT MODE SHIFT
```



```
LOC OBJ                 LINE    SOURCE


E88B 4020100804         1736            DB      CAPS_SHIFT,NUM_SHIFT,SCROLL_SHIFT,ALT_SHIFT,CTL_SHIFT
E890 0201               1737            DB      LEFT_SHIFT,RIGHT_SHIFT
                        1738
                        1739    ;------ SCAN CODE TABLES
                        1740
E892 1BFF00FFFFFF1EFF   1741    K8              DB      27,-1,0,-1,-1,-1,30,-1
E89A FFFFFF1FFF7FFF11   1742                    DB      -1,-1,-1,31,-1,127,-1,17
E8A2 170512141915090F   1743                    DB      23,5,18,20,25,21,9,15
E8AA 101B1D0AFF0113     1744                    DB      16,27,29,10,-1,1,19
E8B1 040607080A0B0CFFFF 1745                    DB      4,6,7,8,10,11,12,-1,-1
E8BA FFFF1C1A18031602   1746                    DB      -1,-1,28,26,24,3,22,2
E8C2 0E0DFFFFFFFFFFFF   1747                    DB      14,13,-1,-1,-1,-1,-1,-1
E8CA 20FF               1748                    DB      ' ',-1
                        1749    ;-------- CTL TABLE SCAN
E8CC                    1750    K9      LABEL   BYTE
E8CC 5E5F606162636465   1751                    DB      94,95,96,97,98,99,100,101
E8D4 6667FFFF77FF84FF   1752                    DB      102,103,-1,-1,119,-1,132,-1
E8DC 73FF74FF75FF76FF   1753                    DB      115,-1,116,-1,117,-1,118,-1
E8E4 FF                 1754                    DB      -1
                        1755    ;-------- LC TABLE
E8E5                    1756    K10     LABEL   BYTE
E8E5 1B                 1757                    DB      01BH,'1234567890-=',08H,09H
E8E6 31323334353637
     3839302D3D0809
E8F4 71776572747975     1758                    DB      'qwertyuiop[]',0DH,-1,'asdfghjkl;',027H
     696F705B5D0DFF
E902 617346666766 6A    1759                    DB      60H,-1,5CH,'zxcvbnm,./',-1,'*',-1,' '
     686C3B27
E900 60FF5C
E910 7A7863766266 6D
     2C2E2FFF2AFF20
E91E FF                 1760                    DB      -1
                        1761
                        1762    ;------ UC TABLE
E91F                    1763    K11     LABEL   BYTE
E91F 1B                 1764                    DB      27,'!@#$',37,05EH,'&*()_+',08H,0
E920 21402324255E
E926 262A28295F2B0800
E92E 51574552545955     1765                    DB      'QWERTYUIOP  ',0DH,-1,'ASDFGHJKL:"'
     494F507B7D0DFF
E93C 4153444647484A
     484C3A22
E947 7EFF               1766                    DB      07EH,-1,' ZXCVBNM<>?',-1,0,-1,' ',-1
E949 7C5A5843564246E
     403C3E3FFF00FF20FF
                        1767    ;------ UC TABLE SCAN
E959                    1768    K12     LABEL   BYTE
E959 54555657585 95A    1769                    DB      84,85,86,87,88,89,90
E960 5B5C5D             1770                    DB      91,92,93
                        1771    ;------ ALT TABLE SCAN
E963                    1772    K13     LABEL   BYTE
E963 68696A6B6C         1773                    DB      104,105,106,107,108
E968 6D6E6F7071         1774                    DB      109,110,111,112,113
                        1775
                        1776    ;------ NUM STATE TABLE
E96D                    1777    K14     LABEL   BYTE
E96D 3738392D343536     1778                    DB      '789-456+1230.'
     2B313233302E
                        1779    ;------ BASE CASE TABLE
E97A                    1780    K15     LABEL   BYTE
E97A 474849FF4BFF4D     1781                    DB      71,72,73,-1,75,-1,77
E9B1 FF4F50515253       1782                    DB      -1,79,80,81,82,83
                        1783
                        1784    ;------ KEYBOARD INTERRUPT ROUTINE
                        1785
E9B7                    1786    KB_INT  PROC    FAR
E9B7 FB                 1787            STI                     ; ALLOW FURTHER INTERRUPTS
E9B8 50                 1788            PUSH    AX
E9B9 53                 1789            PUSH    BX
E9BA 51                 1790            PUSH    CX
E9BB 52                 1791            PUSH    DX
E9BC 56                 1792            PUSH    SI
E9BD 57                 1793            PUSH    DI
E9BE 1E                 1794            PUSH    DS
E9BF 06                 1795            PUSH    ES
E9C0 FC                 1796            CLD                     ; FORWARD DIRECTION
E9C1 B84000         R   1797            MOV     AX,DATA
```

```
LOC OBJ                 LINE    SOURCE

E994 8ED8               1798            MOV     DS,AX           ; SET UP ADDRESSING
E996 E460               1799            IN      AL,KB_DATA      ; READ IN THE CHARACTER
E998 50                 1800            PUSH    AX              ; SAVE IT
E999 E461               1801            IN      AL,KB_CTL       ; GET THE CONTROL PORT
E99B 8AE0               1802            MOV     AH,AL           ; SAVE VALUE
E99D 0C80               1803            OR      AL,80H          ; RESET BIT FOR FETBOARD
E99F E661               1804            OUT     KB_CTL,AL
E9A1 86E0               1805            XCHG    AH,AL           ; GET BACK ORIGINAL CONTROL
E9A3 E661               1806            OUT     KB_CTL,AL       ; KB HAS BEEN RESET
E9A5 58                 1807            POP     AX              ; RECOVER SCAN CODE
E9A6 8AE0               1808            MOV     AH,AL           ; SAVE SCAN CODE IN AH ALSO
                        1809
                        1810    ;------ TEST FOR OVERRUN SCAN CODE FROM KEYBOARD
                        1811
E9A8 3CFF               1812            CMP     AL,0FFH         ; IS THIS AH OVERRUN CHAR
E9AA 7503               1813            JNZ     K16             ; NO, TEST FOR SHIFT KEY
E9AC E97502             1814            JMP     K62             ; BUFFER_FULL_BEEP
                        1815
                        1816    ;------ TEST FOR SHIFT KETS
                        1817
E9AF                    1818    K16:                            ; TEST_SHIFT
E9AF 247F               1819            AND     AL,07FH         ; TURN OFF THE BREAK BIT
E9B1 0E                 1820            PUSH    CS
E9B2 07                 1821            POP     ES              ; ESTABLISH ADDRESS OF SHIFT TABLE
E9B3 BF82EB         R   1822            MOV     DI,OFFSET K6    ; SHIFT KEY TABLE
E9B6 B90800             1823            MOV     CX,K6L          ; LENGTH
E9B9 F2                 1824            REPNE   SCASB           ; LOOK THROUGH THE TABLE FOR A MATCH
E9BA AE
E9BB 8AC4               1825            MOV     AL,AH           ; RECOVER SCAN CODE
E9BD 7403               1826            JE      K17             ; JUMP IF MATCH FOUND
E9BF E98800             1827            JMP     K25             ; IF NO MATCH, THEN SHIFT NOT FOUND
                        1828
                        1829    ;------ SHIFT KEY FOUND
                        1830
E9C2 81EF83EB       R   1831    K17:    SUB     DI,OFFSET K6+1  ; ADJUST PTR TO SCAN CODE MTCH
E9C6 2E8AA583AEB    R   1832            MOV     AH,CS:K7[DI]    ; GET MASK INTO AH
E9CB A880               1833            TEST    AL,80H          ; TEST FOR BREAK KEY
E9CD 7554               1834            JNZ     K23             ; BREAK_SHIFT_FOUND
                        1835
                        1836    ;------ SHIFT MAKE FOUND, DETERMINE SET OR TOGGLE
                        1837
E9CF 80FC10             1838            CMP     AH,SCROLL_SHIFT
E9D2 7307               1839            JAE     K18             ; IF SCROLL SHIFT OR ABOVE, TOGGLE KEY
                        1840
                        1841    ;------ PLAIN SHIFT KEY, SET SHIFT ON
                        1842
E9D4 08261700       R   1843            OR      KB_FLAG,AH      ; TURN ON SHIFT BIT
E9D8 E98300             1844            JMP     K26             ; INTERRUPT_RETURN
                        1845
                        1846    ;------ TOGGLED SHIFT KEY, TEST FOR 1ST MAKE OR NOT
                        1847
E9DB                    1848    K18:                            ; SHIFT-TOGGLE
E9DB F606170004     R   1849            TEST    KB_FLAG, CTL_SHIFT      ; CHECK CTL SHIFT STATE
E9E0 7568               1850            JNZ     K25             ; JUMP IF CTL STATE
E9E2 3C52               1851            CMP     AL, INS_KEY     ; CHECK FOR INSERT KET
E9E4 7525               1852            JNZ     K22             ; JUMP IF NOT INSERT KET
E9E6 F606170008     R   1853            TEST    KB_FLAG, ALT_SHIFT      ; CHECK FOR ALTERNATE SHIFT
E9EB 7403               1854            JZ      K19             ; JUMP IF NOT ALTERNATE SHIFT
E9ED EB5B90             1855            JMP     K25             ; JUMP IF ALTERNATE SHIFT
E9F0 F606170020     R   1856    K19:    TEST    KB_FLAG, NUM_STATE      ; CHECK FOR BASE STATE
E9F5 7508               1857            JNZ     K21             ; JUMP IF NUM LOCK IS ON
E9F7 F606170003     R   1858            TEST    KB_FLAG, LEFT_SHIFT+ RIGHT_SHIFT        ;
E9FC 7400               1859            JZ      K22             ; JUMP IF BASE STATE
                        1860
E9FE                    1861    K20:                            ; NUMERIC ZERO, NOT INSERT KEY
E9FE B83052             1862            MOV     AX, 5230H       ; PUT OUT AN ASCII ZERO
EA01 E90801             1863            JMP     K57             ; BUFFER_FILL
EA04                    1864                                    ; MIGHT BE NUMERIC
EA04 F606170003     R   1865    K21:    TEST    KB_FLAG, LEFT_SHIFT+ RIGHT_SHIFT        ;
EA09 74F3               1866            JZ      K20             ; JUMP NUMERIC, NOT INSERT
                        1867
EA0B                    1868    K22:                            ; SHIFT TOGGLE KEY HIT! PROCESS IT
EA0B 84261800       R   1869            TEST    AH,KB_FLAG_1    ; IS KET ALREADY DEPRESSED
EA0F 7540               1870            JNZ     K26             ; JUMP IF KET ALREADT DEPRESSED
EA11 08261800       R   1871            OR      KB_FLAG_1,AH    ; INDICATE THAT THE KET IS DEPRESSED
EA15 30261700       R   1872            XOR     KB_FLAG,AH      ; TOGGLE THE SHIFT STATE
EA19 3C52               1873            CMP     AL,INS_KEY      ; TEST FOR 1ST MAKE OF INSERT KEY
EA1B 7541               1874            JNE     K26             ; JUMP IF NOT INSERY KET
```



```
LOC OBJ                 LINE    SOURCE

EA1D B80052             1875            MOV     AX,INS_KEY*256          ; SET SCAN CODE INTO AH, 0 INTO AL
EA20 E98901             1876            JMP     K57             ; PUT INTO OUTPUT BUFFER
                        1877
                        1878    ;------ BREAK SHIFT FOUND
                        1879
EA23                    1880    K23:                            ; BREAK-SHIFT-FOUND
EA23 80FC10             1881            CMP     AH,SCROLL_SHIFT ; IS THIS A TOGGLE KEY
EA26 731A               1882            JAE     KZ4             ; YES, HANDLE BREAK TOGGLE
EA28 F6D4               1883            NOT     AH              ; INVERT MASK
EA2A 20261700       R   1884            AND     KB_FLAG,AH      ; TURN OFF SHIFT BIT
EA2E 3C88               1885            CMP     AL,ALT_KEY+80H  ; IS THIS ALTERNATE SHIFT RELEASE
EA30 752C               1886            JNE     K26             ; INTERRUPT_RETURN
                        1887
                        1888    ;------ ALTERNATE SHIFT KET RELEASED, GET THE VALUE INTO BUFFER
                        1889
EA32 A01900         R   1890            MOV     AL,ALT_INPUT
EA35 B400               1891            MOV     AH,0            ; SCAN CODE OF 0
EA37 88261900       R   1892            MOV     ALT_INPUT,AH    ; ZERO OUT THE FIELD
EA3B 3C00               1893            CMP     AL,0            ; WAS THE INPUT=0
EA3D 741F               1894            JE      K26             ; INTERRUPT_RETURN
EA3F E9A301             1895            JMP     K58             ; IT WASN'T, SO PUT IN BUFFER
                        1896
EA42                    1897    K24:                            ; BREAK-TOGGLE
EA42 F6D4               1898            NOT     AH              ; INVERT MASK
EA44 20261800       R   1899            AND     KB_FLAG_1,AH    ; INDICATE NO LONGER DEPRESSED
EA48 EB14               1900            JMP     SHORT K26       ; INTERRUPT_RETURN
                        1901
                        1902    ;------ TEST FOR HOLD STATE
                        1903
EA4A                    1904    K25:                            ; NO-SHIFT-FOUND
EA4A 3C80               1905            CMP     AL,80H          ; TEST FOR BREAK KEY
EA4C 7310               1906            JAE     K26             ; NOTHING FOR BREAK CHARS FROM HERE ON
EA4E F606180008     R   1907            TEST    KB_FLAG_1,HOLD_STATE    ; ARE WE IN HOLD STATE
EA53 7417               1908            JZ      K28             ; BRANCH AROUND TEST IF NOT
EA55 3C45               1909            CMP     AL,NUM_KET
EA57 7405               1910            JE      K26             ; CAN'T END HOLD ON NUM_LOCK
EA59 80261800F7     R   1911            AND     KB_FLAG_1,NOT HOLD_STATE        ; TURN OFF THE HOLD STATE BIT
                        1912
EA5E                    1913    K26:                            ; INTERRUPT-RETURN
EA5E FA                 1914            CLI                     ; TURN OFF INTERRUPTS
EA5F B020               1915            MOV     AL,EOI          ; END OF INTERRUPT COMMAND
EA61 E620               1916            OUT     020H,AL         ; SEND COMMAND TO INTERRUPT CONTROL PORT
EA63                    1917    K27:                            ; INTERRUPT-RETURN-NO-EOI
EA63 07                 1918            POP     ES
EA64 1F                 1919            POP     DS
EA65 5F                 1920            POP     DI
EA66 5E                 1921            POP     SI
EA67 5A                 1922            POP     DX
EA68 59                 1923            POP     CX
EA69 58                 1924            POP     BX
EA6A 58                 1925            POP     AX              ; RESTORE STATE
EA6B CF                 1926            IRET                    ; RETURN, INTERRUPTS BACK ON WITH FLAG CHANGE
                        1927
                        1928    ;------ NOT IN  HOLD STATE, TEST FOR SPECIAL CHARS
                        1929
EA6C                    1930    K28:                            ; NO-HOLD-STATE
EA6C F606170008     R   1931            TEST    KB_FLAG,ALT_SHIFT       ; ARE WE IN ALTERNATE SHIFT
EA71 7503               1932            JNZ     K29             ; JUMP IF ALTERNATE SHIFT
EA73 E98F00             1933            JMP     K38             ; JUMP IF NOT ALTERNATE
                        1934
                        1935    ;------ TEST FOR RESET KET SEQUENCE (CTL ALT DEL)
                        1936
EA76                    1937    K29:                            ; TEST-RESET
EA76 F606170004     R   1938            TEST    KB_FLAG,CTL_SHIFT       ; ARE WE IN CONTROL SHIFT ALSO
EA7B 7431               1939            JZ      K31             ; NO_RESET
EA7D 3C53               1940            CMP     AL,DEL_KEY      ; SHIFT STATE IS THERE, TEST KET
EA7F 752D               1941            JNE     K31             ; NO_RESET
                        1942
                        1943    ;------ CTL-ALT-OEL HAS BEEN FOUND, DO I/O CLEANUP
                        1944
EA81 C706720034 12  R   1945            MOV     RESET_FLAG, 1234H       ; SET FLAG FOR RESET FUNCTION
EA87 E901F5             1946            JMP     RESET           ; JUMP TO POWER ON DIAGNOSTICS
                        1947
                        1948    ;------ ALT-INPUT-TABLE
EA8A                    1949    K30     LABEL   BYTE
EA8A 52 4F 50 51 4B 4C 4D 50    1950            DB      82,79,80,81,75,76,77
```

```
LOC OBJ                 LINE    SOURCE

EA91 474849             1951            DB      71,72,73                ; 10 NUMBERS ON KEYPAD
                        1952    ;------ SUPER-SHIFT-TABLE
EA94 1011121314151617   1953            DB      16,17,18,19,20,21,22,23 ; A-Z TYPEWRITER CHARS
EA9C 18191E1F20212223   1954            DB      24,25,30,31,32,33,34,35
EAA4 2425262C2D2E2F30   1955            DB      36,37,38,44,45,46,47,48
EAAC 3132               1956            DB      49,50
                        1957
                        1958    ;------ IN ALTERNATE SHIFT, RESET NOT FOUND
                        1959
EAAE                    1960    K31:                                    ; NO-RESET
EAAE 3C39               1961            CMP     AL,57                   ; TEST FOR SPACE KEY
EAB0 7505               1962            JNE     K32                     ; NOT THERE
EAB2 B020               1963            MOV     AL,' '                  ; SET SPACE CHAR
EAB4 E92501             1964            JMP     K57                     ; BUFFER_FILL
                        1965
                        1966    ;------ LOOK FOR KEY PAD ENTRY
                        1967
EAB7                    1968    K32:                                    ; ALT-KEY-PAD
EAB7 BF8AEA         R   1969            MOV     DI,OFFSET K30           ; ALT-INPUT-TABLE
EABA B90A00             1970            MOV     CX,10                   ; LOOK FOR ENTRY USING KEYPAD
EABD F2                 1971            REPNE   SCASB                   ; LOOK FOR MATCH
EABE AE
EABF 7512               1972            JNE     K33                     ; NO_ALT_KEYPAD
EAC1 81EF8BEA       R   1973            SUB     DI,OFFSET K30+1         ; DI NOW HAS ENTRY VALUE
EAC5 A01900         R   1974            MOV     AL,ALT_INPUT            ; GET THE CURRENT BYTE
EAC8 B40A               1975            MOV     AH,10                   ; MULTIPLT BY 10
EACA F6E4               1976            MUL     AH
EACC 03C7               1977            ADD     AX,DI                   ; ADD IN THE LATEST ENTRY
EACE A21900         R   1978            MOV     ALT_INPUT,AL            ; STORE IT AWAY
EAD1 EB8B               1979            JMP     K26                     ; THROW AWAY THAT KEYSTROKE
                        1980
                        1981    ;------ LOOK FOR SUPERSHIFT ENTRY
                        1982
EAD3                    1983    K33:                                    ; NO-ALT-KEYPAD
EAD3 C606190000     R   1984            MOV     ALT_INPUT,0             ; ZERO ANY PREVIOUS ENTRY INTO INPUT
EAD8 B91A00             1985            MOV     CX,26                   ; DI;ES ALREADY POINTING
EADB F2                 1986            REPNE   SCASB                   ; LOOK FOR MATCH IN ALPHABET
EADC AE
EADD 7505               1987            JNE     K34                     ; NOT FOUND, FUNCTION KEY OR OTHER
EADF B000               1988            MOV     AL,0                    ; ASCII CODE OF ZERO
EAE1 E9F800             1989            JMP     K57                     ; PUT IT IN THE BUFFER
                        1990
                        1991    ;------ LOOK FOR TOP ROW OF ALTERNATE SHIFT
                        1992
EAE4                    1993    K34:                                    ; ALT-TOP-ROW
EAE4 3C02               1994            CMP     AL,2                    ; KEY WITH '1' ON IT
EAE6 720C               1995            JB      K35                     ; NOT ONE OF INTERESTING KETS
EAE8 3C0E               1996            CMP     AL,14                   ; IS IT IN THE REGION
EAEA 7308               1997            JAE     K35                     ; ALT-FUNCTION
EAEC 80C476             1998            ADD     AH,118                  ; CONVERT PSUEDO SCAN CODE TO RANGE
EAEF B000               1999            MOV     AL,0                    ; INDICATE AS SUCH
EAF1 E9E800             2000            JMP     K57                     ; BUFFER_FILL
                        2001
                        2002    ;------ TRANSLATE ALTERNATE SHIFT PSEUDO SCAN CODES
                        2003
EAF4                    2004    K35:                                    ; ALT-FUNCTION
EAF4 3C3B               2005            CMP     AL,59                   ; TEST FOR IN TABLE
EAF6 7303               2006            JAE     K37                     ; ALT-CONTINUE
EAF8                    2007    K36:                                    ; CLOSE-RETURN
EAF8 E963FF             2008            JMP     K26                     ; IGNORE THE KEY
EAFB                    2009    K37:                                    ; ALT-CONTINUE
EAFB 3C47               2010            CMP     AL,71                   ; IN KEYPAD REGION.
EAFD 73F9               2011            JAE     K36                     ; IF SO, IGNORE
EAFF 8B63E9         R   2012            MOV     BX,OFFSET K13           ; ALT SHIFT PSEUDO SCAN TABLE
EB02 E92501             2013            JMP     K63                     ; TRANSLATE THAT
                        2014
                        2015    ;------ NOT IN ALTERNATE SHIFT
                        2016
EB05                    2017    K38:                                    ; NOT-ALT-SHIFT
EB05 F606170004     R   2018            TEST    KB_FLAG,CTL_SHIFT       ; ARE WE IN CONTROL SHIFT
EB0A 7458               2019            JZ      K44                     ; NOT-CTL-SHIFT
                        2020
                        2021    ;------ CONTROL SHIFT, TEST SPECIAL CHARACTERS
                        2022    ;------ TEST FOR BREAK AND PAUSE KETS
                        2023
EB0C 3C46               2024            CMP     AL,SCROLL_KEY           ; TEST FOR BREAK
EB0E 7518               2025            JNE     K39                     ; NO-BREAK
```



```
LOC OBJ                 LINE    SOURCE

EB10 BB1E00         R   2026            MOV     BX,OFFSET KB_BUFFER     ; RESET BUFFER TO EMPTT
EB13 891E1A00       R   2027            MOV     BUFFER_HEAD,BX          ;
EB17 891E1C00       R   2028            MOV     BUFFER_TAIL,BX          ;
EB1B C606710080     R   2029            MOV     BIOS_BREAK,80H          ; TURN ON BIOS_BREAK BIT
EB20 CD1B               2030            INT     1BH                     ; BREAK INTERRUPT VECTOR
EB22 B80000             2031            MOV     AX, 0                   ; PUT OUT DUMMY CHARACTER
EB25 E9B400             2032            JMP     K57                     ; BUFFER_FILL
                        2033
EB28                    2034    K39:                                    ; NO-BREAK
EB28 3C45               2035            CMP     AL,NUM_KEY              ; LOOK FOR PAUSE KEY
EB2A 7521               2036            JNE     K41                     ; NO-PAUSE
EB2C 800E180008     R   2037            OR      KB_FLAG_1,HOLD_STATE    ; TURN ON THE HOLD FLAG
EB31 B020               2038            MOV     AL,EOI                  ; END OF INTERRUPT TO CONTROL PORT
EB33 E620               2039            OUT     020H,AL                 ; ALLOW FURTHER KETSTROKE INTS
                        2040
                        2041    ;------ DURING PAUSE INTERVAL, TURN CRT BACK ON
                        2042
EB35 803E490007     R   2043            CMP     CRT_MODE,7              ; IS THIS BLACK AND WHITE CARD
EB3A 7407               2044            JE      K40                     ; YES, NOTHING TO DO
EB3C BAD803             2045            MOV     DX,03D8H                ; PORT FOR COLOR CARD
EB3F A06500         R   2046            MOV     AL,CRT_MODE_SET         ; GET THE VALUE OF THE CURRENT MODE
EB42 EE                 2047            OUT     DX,AL                   ; SET THE CRT MODE, SO THAT CRT IS ON
EB43                    2048    K40:                                    ; PAUSE-LOOP
EB43 F606180008     R   2049            TEST    KB_FLAG_1,HOLD_STATE
EB48 75F9               2050            JNZ     K40                     ; LOOP UNTIL FLAG TURNED OFF
EB4A E916FF             2051            JMP     K27                     ; INTERRUPT_RETURN_NO_EOI
EB4D                    2052    K41:                                    ; NO-PAUSE
                        2053
                        2054    ;------ TEST SPECIAL CASE KEY 55
                        2055
EB4D 3C37               2056            CMP     AL,55
EB4F 7506               2057            JNE     K42                     ; NOT-KEY-55
EB51 B80072             2058            MOV     AX,114*256              ; START/STOP PRINTING SWITCH
EB54 E98500             2059            JMP     K57                     ; BUFFER_FILL
                        2060
                        2061    ;------ SET UP TO TRANSLATE CONTROL SHIFT
                        2062
EB57                    2063    K42:                                    ; NOT-KEY-55
EB57 BB92E8         R   2064            MOV     BX,OFFSET K8            ; SET UP TO TRANSLATE CTL
EB5A 3C3B               2065            CMP     AL,59                   ; IS IT IN TABLE
EB5C 7303               2066            JAE     K43                     ; CTL-TABLE-TRANSLATE
EB5E EB7890             2067            JMP     K56                     ; YES, GO TRANSLATE CHAR
EB61                    2068    K43:                                    ; CTL-TABLE-TRANSLATE
EB61 BBCCE8         R   2069            MOV     BX,OFFSET K9            ; CTL TABLE SCAN
EB64 E9C300             2070            JMP     K63                     ; TRANSLATE_SCAN
                        2071
                        2072    ;------ NOT IN CONTROL SHIFT
                        2073
EB67                    2074    K44:                                    ; NOT-CTL-SHIFT
                        2075
EB67 3C47               2076            CMP     AL,71                   ; TEST FOR KEYPAD REGION
EB69 732D               2077            JAE     K48                     ; HANDLE KEYPAD REGION
EB6B F606170003     R   2078            TEST    KB_FLAG,LEFT_SHIFT+RIGHT_SHIFT
EB70 7458               2079            JZ      K54                     ; TEST FOR SHIFT STATE
                        2080
                        2081    ;------ UPPER CASE, HANDLE SPECIAL CASES
                        2082
EB72 3C0F               2083            CMP     AL,15                   ; BACK TAB KEY
EB74 7506               2084            JNE     K45                     ; NOT-BACK-TAB
EB76 B8000F             2085            MOV     AX,15*256               ; SET PSEUDO SCAN CODE
EB79 EB6190             2086            JMP     K57                     ; BUFFER_FILL
                        2087
EB7C                    2088    K45:                                    ; NOT-BACK-TAB
EB7C 3C37               2089            CMP     AL,55                   ; PRINT SCREEN KEY
EB7E 7509               2090            JNE     K46                     ; NOT-PRINT-SCREEN
                        2091
                        2092    ;------ ISSUE INTERRUPT TO INDICATE PRINT SCREEN FUNCTION
                        2093
EB80 B020               2094            MOV     AL,EOI                  ; END OF CURRENT INTERRUPT
EB82 E620               2095            OUT     020H,AL                 ;  SO FURTHER THINGS CAN HAPPEN
EB84 CD05               2096            INT     5H                      ; ISSUE PRINT SCREEN INTERRUPT
EB86 E9DAFE             2097            JMP     K27                     ; GO BACK WITHOUT EOI OCCURRING
                        2098
EB89                    2099    K46:                                    ; NOT-PRINT-SCREEN
EB89 3C3B               2100            CMP     AL,59                   ; FUNCTION KETS
EB8B 7206               2101            JB      K47                     ; NOT-UPPER-FUNCTION
EB8D BB59E9         R   2102            MOV     BX,OFFSET K12           ; UPPER CASE PSEUDO SCAN CODES
```

```
LOC OBJ                 LINE    SOURCE

E890 E99700             2103            JMP     K63                     ; TRANSLATE_SCAN
                        2104
E893                    2105    K47:                                    ; NOT-UPPER-FUNCTION
E893 BB1FE9       R     2106            MOV     BX,OFFSEY K11           ; POINT TO UPPER CASE TABLE
E896 EB40               2107            JMP     SHORT K56               ; OK, TRANSLATE THE CHAR
                        2108
                        2109    ;------ KETPAD KEYS, MUST TEST NUM LOCK FOR DETERMINATION
                        2110
E898                    2111    K48:                                    ; KETPAD-REGION
E898 F606170020   R     2112            TEST    KB_FLAG,NUM_STATE       ; ARE WE IN NUM_LOCK
E89D 7520               2113            JNZ     K52                     ; TEST FOR SURE
E89F F606170003   R     2114            YEST    KB_FLAG,LEFT_SHIFT+RIGHT_SHIFT  ; ARE WE IN SHIFT STATE
E8A4 7520               2115            JNZ     K53                     ; IF SHIFTED, REALLT NUM STATE
                        2116
                        2117    ;------ BASE CASE FOR KETPAD
                        2118
E8A6                    2119    K49:                                    ; BASE-CASE
                        2120                                            ;
E8A6 3C4A               2121            CMP     AL,74                   ; SPECIAL CASE FOR A COUPLE OF KEYS
E8A8 740B               2122            JE      K50                     ; MINUS
E8AA 3C4E               2123            CMP     AL,78
E8AC 740C               2124            JE      K51
E8AE 2C47               2125            SUB     AL,71                   ; CONVERT ORIGIN
E8B0 B87AE9       R     2126            MOV     BX,OFFSET K15           ; BASE CASE TABLE
E8B3 EB77               2127            JMP     SHORT K64               ; CONVERT TO PSEUDO SCAN
                        2128
E8B5 B8D04A             2129    K50:    MOV     AX,74*256+'-'           ; MINUS
E8B8 EB22               2130            JMP     SHORT K57               ; BUFFER_FILL
                        2131
E8BA B82B4E             2132    K51:    MOV     AX,78*256+'+'           ; PLUS
E8BD EB1D               2133            JMP     SHORT K57               ; BUFFER_FILL
                        2134
                        2135    ;------ MIGHT BE NUM LOCK, TEST SHIFT STATUS
                        2136
E8BF                    2137    K52:                                    ; ALMOST-NUM-STATE
E8BF F606170003   R     2138            TEST    KB_FLAG,LEFT_SHIFT+RIGHT_SHIFT
E8C4 75E0               2139            JNZ     K49                     ; SHIFTED TEMP OUT OF NUM STATE
                        2140
E8C6                    2141    K53:                                    ; REALLT_NUM_STATE
E8C6 2C46               2142            SUB     AL,70                   ; CONVERT ORIGIN
E8C8 B86DE9       R     2143            MOV     BX,OFFSET K14           ; NUM STATE TABLE
E8CB EB0B               2144            JMP     SHORT K56               ; TRANSLATE_CHAR
                        2145
                        2146    ;------ PLAIN OLD LOWER CASE
                        2147
E8CD                    2148    K54:                                    ; NOT-SHIFT
E8CD 3C3B               2149            CMP     AL,59                   ; TEST FOR FUNCTION KEYS
E8CF 7204               2150            JB      K55                     ; NOT-LOWER-FUNCTIOM
E8D1 B000               2151            MOV     AL,0                    ; SCAN CODE IN AH ALREADY
E8D3 EB07               2152            JMP     SHORT K57               ; BUFFER_FILL
                        2153
E8D5                    2154    K55:                                    ; NOT-LOWER-FUNCTION
E8D5 B8E5E8       R     2155            MOV     BX,OFFSET K10           ; LC TABLE
                        2156
                        2157    ;------ TRANSLATE THE CHARACTER
                        2158
E8D8                    2159    K56:                                    ; TRANSLATE-CHAR
E8D8 FEC8               2160            DEC     AL                      ; CONVERT ORIGIN
E8DA 2E07               2161            XLAT    CS:K11                  ; CONVERT THE SCAN CODE TO ASCII
                        2162
                        2163    ;------ PUT CHARACTER INTO BUFFER
                        2164
E8DC                    2165    K57:                                    ; BUFFER-FILL
E8DC 3CFF               2166            CMP     AL,-1                   ; IS THIS AN IGNORE CHAR
E8DE 741F               2167            JE      K59                     ; YES, DO NOTHING WITH IT
E8E0 80FCFF             2168            CMP     AH,-1                   ; LOOK FOR -1 PSEUDO SCAN
E8E3 741A               2169            JE      K59                     ; HEAR_INTERRUPT_RETURN
                        2170
                        2171    ;------ HANDLE THE CAPS LOCK PROBLEM
                        2172
E8E5                    2173    K58:                                    ; BUFFER-FILL-NOTEST
E8E5 F606170040   R     2174            TEST    KB_FLAG,CAPS_STATE      ; ARE WE IN CAPS LOCK STATE
E8EA 7420               2175            JZ      K61                     ; SKIP IF NOT
                        2176
                        2177    ;------ IN CAPS LOCK STATE
                        2178
E8EC F606170003   R     2179            TEST    KB_FLAG,LEFT_SHIFT+RIGHT_SHIFT  ; TEST FOR SHIFT STATE
```



```
LOC OBJ                 LINE    SOURCE

E8F1 740F               2180            JZ      K60                     ; IF NOT SHIFT, CONVERT LOWER TO UPPER
                        2181
                        2182    ;------ CONVERT ANY UPPER CASE TO LOWER CASE
                        2183
E8F3 3C41               2184            CMP     AL,'A'                  ; FIND OUT IF ALPHABETIC
E8F5 7215               2185            JB      K61                     ; NOT_CAPS_STATE
E8F7 3C5A               2186            CMP     AL,'Z'
E8F9 7711               2187            JA      K61                     ; NOT_CAPS_STATE
E8FB 0420               2188            ADD     AL,'a'-'A'              ; CONVERT TO LOWER CASE
E8FD EB00               2189            JMP     SHORT K61               ; NOT_CAPS_STATE
                        2190
E8FF                    2191    K59:                                    ; HEAR-INTERRUPT-RETURN
E8FF E95CFE             2192            JMP     K26                     ; INTERRUPT_RETURN
                        2193
                        2194    ;------ CONVERT ANY LOWER CASE TO UPPER CASE
                        2195
EC02                    2196    K60:                                    ; LOWER-TO-UPPER
EC02 3C61               2197            CMP     AL,'a'                  ; FIND OUT IF ALPHABETIC
EC04 7206               2198            JB      K61                     ; NOT_CAPS_STATE
EC06 3C7A               2199            CMP     AL,'z'
EC08 7702               2200            JA      K61                     ; NOT_CAPS_STATE
EC0A 2C20               2201            SUB     AL,'a'-'A'              ; CONVERT TO UPPER CASE
                        2202
EC0C                    2203    K61:                                    ; NOT-CAPS-STATE
EC0C 8B1E1C00     R     2204            MOV     BX,BUFFER_TAIL          ; GET THE END POINTER TO THE BUFFER
EC10 8BF3               2205            MOV     SI,BX                   ; SAVE THE VALUE
EC12 EB60FC             2206            CALL    K4                      ; ADVANCE THE TAIL
EC15 3B1E1A00     R     2207            CMP     BX,BUFFER_HEAD          ; HAS THE BUFFER WRAPPED AROUND
EC19 7409               2208            JE      K62                     ; BUFFER_FULL_BEEP
EC1B 8904               2209            MOV     [SI],AX                 ; STORE THE VALUE
EC1D 891E1C00     R     2210            MOV     BUFFER_TAIL,BX          ; MOVE THE POINTER UP
EC21 E93AFE             2211            JMP     K26                     ; INTERRUPT_RETURN
                        2212
                        2213    ;------ BUFFER IS FULL, SOUND THE BEEPER
                        2214
EC24                    2215    K62:                                    ; BUFFER-FULL-BEEP
EC24 E80D00             2216            CALL    ERROR_BEEP
EC27 E934FE             2217            JMP     K26                     ; INTERRUPT_RETURN
                        2218
                        2219    ;------ TRANSLATE SCAN FOR PSEUDO SCAN CODES
                        2220
EC2A                    2221    K63:                                    ; TRANSLATE-SCAN
EC2A 2C3B               2222            SUB     AL,59                   ; CONVERT ORIGIN TO FUNCTION KEYS
EC2C                    2223    K64:                                    ; TRANSLATE-SCAN-ORGD
EC2C 2ED7               2224            XLAT    CS:K9                   ; CTL TABLE SCAN
EC2E 8AE0               2225            MOV     AH,AL                   ; PUT VALUE INTO AH
EC30 B000               2226            MOV     AL,0                    ; ZERO ASCII CODE
EC32 EBA8               2227            JMP     K57                     ; PUT IT INTO THE BUFFER
                        2228
                        2229    KB_INT  ENDP
EC34                    2230    ERROR_BEEP      PROC    NEAR
EC34 50                 2231            PUSH    AX              ; SAVE REGISTERS
EC35 53                 2232            PUSH    BX
EC36 51                 2233            PUSH    CX
EC37 B9C000             2234            MOV     BX,0C0H         ; NUMBER OF CYCLES FOR 1/8 SECOND TONE
EC3A E461               2235            IN      AL,KB_CTL       ; GET CONTROL INFORMATION
EC3C 50                 2236            PUSH    AX              ; SAVE
EC3D                    2237    K65:                            ; BEEP-CYCLE
EC3D 24FC               2238            AND     AL,0FCH         ; TURN OFF TIMER GATE AND SPEAKER DATA
EC3F E661               2239            OUT     KB_CTL,AL       ; OUTPUT TO CONTROL
EC41 B94800             2240            MOV     CX,48H          ; HALF CYCLE TIME FOR TONE
EC44 E2FE               2241    K66:    LOOP    K66             ; SPEAKER OFF
EC46 0C02               2242            OR      AL,2            ; TURN ON SPEAKER BIT
EC48 E661               2243            OUT     KB_CTL,AL       ; OUTPUT TO CONTROL
EC4A B94800             2244            MOV     CX,48H          ; SET UP COUNT
EC4D E2FE               2245    K67:    LOOP    K67             ; ANOTHER HALF CYCLE
EC4F 4B                 2246            DEC     BX              ; TOTAL TIME COUNT
EC50 75EB               2247            JNZ     K65             ; DO ANOTHER CYCLE
EC52 58                 2248            POP     AX              ; RECOVER CONTROL
EC53 E661               2249            OUT     KB_CTL,AL       ; OUTPUT THE CONTROL
EC55 59                 2250            POP     CX              ; RECOVER REGISTERS
EC56 5B                 2251            POP     BX
EC57 58                 2252            POP     AX
EC58 C3                 2253            RET
                        2254    ERROR_BEEP      ENDP
```

```
LOC OBJ                 LINE    SOURCE


                        2255    ;-- INT 13 --------------------------------
                        2256    ;DISKETTE I/O
                        2257    ;       THIS INTERFACE PROVIDES ACCESS TO THE 5 1/4" DISKETTE DRIVES
                        2258    ;INPUT
                        2259    ;       (AH)=0  RESET DISKETTE SYSTEM
                        2260    ;               HARD RESET TO NEC, PREPARE COMMAND, RECAL REQD ON ALL DRIVES
                        2261    ;       (AH)=1  READ THE STATUS OF THE SYSTEM INTO (AL)
                        2262    ;               DISKETTE_STATUS FROM LAST OPN IS USED
                        2263    ;       REGISTERS FOR READ/WRITE/VERIFY/FORMAT
                        2264    ;       (DL) - DRIVE NUMBER (0-3 ALLOWED, VALUE CHECKED)
                        2265    ;       (DH) - HEAD NUMBER (0-1 ALLOWED, NOT VALUE CHECKED)
                        2266    ;       (CH) - TRACK NUMBER (0-39, NOT VALUE CHECKED)
                        2267    ;       (CL) - SECTOR NUMBER (1-8, NOT VALUE CHECKED)
                        2268    ;       (AL) - NUMBER OF SECTORS ( MAX = 8, NOT VALUE CHECKED)
                        2269    ;
                        2270    ;       (ES:BX) - ADDRESS OF BUFFER ( NOT REQUIRED FOR VERIFY)
                        2271    ;
                        2272    ;       (AH)=2  READ THE DESIRED SECTORS INTO MEMORY
                        2273    ;       (AH)=3  WRITE THE DESIRED SECTORS FROM MEMORY
                        2274    ;       (AH)=4  VERIFY THE DESIRED SECTORS
                        2275    ;       (AH)=5  FORMAT THE DESIRED TRACK
                        2276    ;               FOR THE FORMAT OPERATION, THE BUFFER POINTER (ES:BX) MUST
                        2277    ;               POINT TO THE COLLECTION OF DESIRED ADDRESS FIELDS FOR THE
                        2278    ;               TRACK.  EACH FIELD IS COMPOSED OF 4 BYTES, (C,H,R,N), WHERE
                        2279    ;               C = TRACK NUMBER, H=HEAD NUMBER, R = SECTOR NUMBER, N= NUMBER
                        2280    ;               OF BYTES PER SECTOR (00=128, 01=256, 02=512, 03=1024,)
                        2281    ;               THERE MUST BE ONE ENTRY FOR EVERY SECTOR ON THE TRACK.
                        2282    ;               THIS INFORMATION IS USED TO FIND THE REQUESTED SECTOR DURING
                        2283    ;               READ/WRITE ACCESS.
                        2284    ; DATA VARIABLE -- DISK_POINTER
                        2285    ;       DOUBLE WORD POINTER TO THE CURRENT SET OF DISKETTE PARAMETERS
                        2286    ; OUTPUT
                        2287    ;       AH = STATUS OF OPERATION
                        2288    ;               STATUS BITS ARE DEFINED IN THE EQUATES FOR DISKETTE_STATUS
                        2289    ;               VARIABLE IN THE DATA SEGMENT OF THIS MODULE
                        2290    ;       CY = 0  SUCCESSFUL OPERATION (AH=0 ON RETURN)
                        2291    ;       CY = 1  FAILED OPERATION (AH HAS ERROR REASON)
                        2292    ;       FOR READ/WRITE/VERIFY
                        2293    ;               DS,BX,DX,CH,CL PRESERVED
                        2294    ;               AL = NUMBER OF SECTORS ACTUALLY READ
                        2295    ;               ***** AL MAY NOT BE CORRECT IF TIME OUT ERROR OCCURS
                        2296    ;       NOTE: IF AN ERROR IS REPORTED BY THE DISKETTE CODE, THE APPROPRIATE
                        2297    ;               ACTION IS TO RESET THE DISKETTE, THEN RETRY THE OPERATION.
                        2298    ;               ON READ ACCESSES, NO MOTOR START DELAY IS TAKEN, SO THAT
                        2299    ;               THREE RETRIES ARE REQUIRED ON READS TO ENSURE THAT THE
                        2300    ;               PROBLEM IS NOT DUE TO MOTOR START-UP.
                        2301    ;----------------------------------------
                        2302            ASSUME  CS:CODE,DS:DATA,ES:DATA
EC59                    2303    DISKETTE_IO     PROC    FAR
EC59 FB                 2304            STI                     ; INTERRUPTS BACK ON
EC5A 53                 2305            PUSH    BX              ; SAVE ADDRESS
EC5B 51                 2306            PUSH    CX
EC5C 1E                 2307            PUSH    DS              ; SAVE SEGMENT REGISTER VALUE
EC5D 56                 2308            PUSH    SI              ; SAVE ALL REGISTERS DURING OPERATION
EC5E 57                 2309            PUSH    DI
EC5F 55                 2310            PUSH    BP
EC60 52                 2311            PUSH    DX
EC61 8BEC               2312            MOV     BP,SP           ; SET UP POINTER TO HEAD PARM
EC63 8E4000         R   2313            MOV     SI,DATA
EC66 8EDE               2314            MOV     DS,SI           ; SET DATA REGION
EC68 E81C00             2315            CALL    JI              ; CALL THE REST TO ENSURE DS RESTORED
EC6B BB0400             2316            MOV     BX,4            ; GET THE MOTOR WAIT PARAMETER
EC6E E8FF01             2317            CALL    GET_PARM
EC71 88264000       R   2318            MOV     MOTOR_COUNT,AH  ; SET THE TIMER COUNT FOR THE MOTOR
EC75 8A264100       R   2319            MOV     AH,DISKETTE_STATUS      ; GET STATUS OF OPERATION
EC79 80FC01             2320            CMP     AH,1            ; SET THE CARRY FLAG TO INDICATE
EC7C F5                 2321            CMC                     ;  SUCCESS OR FAILURE
EC7D 5A                 2322            POP     DX              ; RESTORE ALL REGISTERS
EC7E 5D                 2323            POP     BP
EC7F 5F                 2324            POP     DI
EC80 5E                 2325            POP     SI
EC81 1F                 2326            POP     DS
EC82 59                 2327            POP     CX
EC83 5B                 2328            POP     BX              ; RECOVER ADDRESS
EC84 CA0200             2329            RET     2               ; THROW AWAY SAVED FLAGS
                        2330    DISKETTE_IO     ENDP
EC87                    2331    J1      PROC    NEAR
```



```
LOC OBJ                 LINE    SOURCE


EC87 8AF0               2332            MOV     DH,AL           ; SAVE # SECTORS IN DH
EC89 80263F007F     R   2333            AND     MOTOR_STATUS,07FH       ; INDICATE A READ OPERATION
EC8E 0AE4               2334            OR      AH,AH           ; AH=0
EC90 7427               2335            JZ      DISK_RESET
EC92 FECC               2336            DEC     AH              ; AH=1
EC94 7474               2337            JZ      DISK_STATUS
EC96 C606410000     R   2338            MOV     DISKETTE_STATUS,0       ; RESET THE STATUS INDICATOR
EC9B 80FA04             2339            CMP     DL,4            ; TEST FOR DRIVE IN 0-3 RANGE
EC9E 7313               2340            JAE     J3              ; ERROR IF ABOVE
ECA0 FECC               2341            DEC     AH              ; AH=2
ECA2 746A               2342            JZ      DISK_READ
ECA4 FECC               2343            DEC     AH              ; AH=3
ECA6 7503               2344            JNZ     J2              ; TEST_DISK_VERF

ECA8 E99600             2345            JMP     DISK_WRITE
ECAB                    2346    J2:                             ; TEST_DISK_VERF

ECAB FECC               2347            DEC     AH              ; AH=4
ECAD 7468               2348            JZ      DISK_VERF
ECAF FECC               2349            DEC     AH              ; AH=5
ECB1 7468               2350            JZ      DISK_FORMAT
ECB3                    2351    J3:                             ; BAD_COMMAND
ECB3 C606410001     R   2352            MOV     DISKETTE_STATUS,BAD_CMD ; ERROR CODE, NO SECTORS TRANSFERRED
ECB8 C3                 2353            RET                     ; UNDEFINED OPERATION
                        2354    J1      ENDP
                        2355
                        2356    ;------ RESET THE DISKETTE SYSTEM
                        2357
ECB9                    2358    DISK_RESET      PROC    NEAR
ECB9 BAF203             2359            MOV     DX,03F2H        ; ADAPTER CONTROL PORT
ECBC FA                 2360            CLI                     ; NO INTERRUPTS
ECBD A03F00         R   2361            MOV     AL,MOTOR_STATUS ; WHICH MOTOR IS ON
ECC0 B104               2362            MOV     CL,4            ; SHIFT COUNT
ECC2 D2E0               2363            SAL     AL,CL           ; MOVE MOTOR VALUE TO HIGH NYBBLE
ECC4 A820               2364            TEST    AL, 20H         ; SELECT CORRESPONDING DRIVE
ECC6 750C               2365            JNZ     J5              ; JUMP IF MOTOR ONE IS ON
ECC8 A840               2366            TEST    AL, 40H
ECCA 7506               2367            JNZ     J4              ; JUMP IF MOTOR TWO IS ON
ECCC A880               2368            TEST    AL, 80H
ECCE 7406               2369            JZ      J6              ; JUMP IF MOTOR ZERO IS ON
ECD0 FEC0               2370            INC     AL
ECD2 FEC0               2371    J4:     INC     AL
ECD4 FEC0               2372    J5:     INC     AL
ECD6 0C08               2373    J6:     OR      AL,8            ; TURN ON INTERRUPT ENABLE
ECD8 EE                 2374            OUT     DX,AL           ; RESET THE ADAPTER
ECD9 C6063E0000     R   2375            MOV     SEEK_STATUS,0   ; SET RECAL REQUIRED ON ALL DRIVES
ECDE C606410000     R   2376            MOV     DISKETTE_STATUS,0       ; SET OK STATUS FOR DISKETTE
ECE3 0C04               2377            OR      AL,4            ; TURN OFF RESET
ECE5 EE                 2378            OUT     DX,AL           ; TURN OFF THE RESET
ECE6 FB                 2379            STI                     ; REENABLE THE INTERRUPTS
ECE7 E82802             2380            CALL    CHK_STAT_2      ; DO SENSE INTERRUPT STATUS FOLLOWING RESET
ECEA A04200         R   2381            MOV     AL,NEC_STATUS   ; IGNORE ERROR RETURN AND DO OWN TEST
ECED 3CC0               2382            CMP     AL,0C0H         ; TEST FOR DRIVE READY TRANSITION
ECEF 7407               2383            JZ      J7              ; EVERYTHING OK
ECF1 800E410020     R   2384            OR      DISKETTE_STATUS,BAD_NEC ; SET ERROR CODE
ECF6 EB11               2385            JMP     SHORT J8                ; RESET_RET
                        2386
                        2387    ;------ SEND SPECIFY COMMAND TO NEC
                        2388
ECF8                    2389    J7:                             ; DRIVE_READY
ECF8 B403               2390            MOV     AH,03H          ; SPECIFY COMMAND
ECFA E84701             2391            CALL    NEC_OUTPUT      ; OUTPUT THE COMMAND
ECFD BB0100             2392            MOV     BX,1            ; FIRST BYTE PARM IN BLOCK
ED00 E86D01             2393            CALL    GET_PARM        ;  TO THE NEC CONTROLLER
ED03 BB0300             2394            MOV     BX,3            ; SECOND BYTE PARM IN BLOCK
ED06 E86701             2395            CALL    GET_PARM        ;  TO THE NEC CONTROLLER
ED09                    2396    J8:                             ; RESET_RET
ED09 C3                 2397            RET                     ; RETURN TO CALLER
                        2398    DISK_RESET      ENDP
                        2399
                        2400    ;------ DISKETTE STATUS ROUTINE
                        2401
ED0A                    2402    DISK_STATUS     PROC    NEAR
ED0A A04100         R   2403            MOV     AL,DISKETTE_STATUS
ED0D C3                 2404            RET
                        2405    DISK_STATUS     ENDP
                        2406
```

```
LOC OBJ                 LINE    SOURCE

                        2407    ;------ DISKETTE READ
                        2408
E00E                    2409    DISK_READ       PROC    NEAR
E00E B046               2410            MOV     AL,046H         ; READ COMMAND FOR DMA
E010                    2411    J9:                             ; DISK_READ_CONT
E010 E8B901             2412            CALL    DMA_SETUP       ; SET UP THE DMA
E013 B466               2413            MOV     AH,066H         ; SET UP READ COMMAND FOR NEC CONTROLLER
E015 EB36               2414            JMP     SHORT RW_OPN    ; GO DO THE OPERATION
                        2415    DISK_READ       ENDP
                        2416
                        2417    ;------ DISKETTE VERIFY
                        2418
E017                    2419    DISK_VERF       PROC    NEAR
E017 B042               2420            MOV     AL,042H                 ; VERIFY COMMAND FOR DMA
E019 EBF5               2421            JMP     J9                      ; DO AS IF DISK READ
                        2422    DISK_VERF       ENDP
                        2423
                        2424    ;------ DISKETTE FORMAT
                        2425
E01B                    2426    DISK_FORMAT     PROC    NEAR
E01B 800E3F0080     R   2427            OR      MOTOR_STATUS,80H        ; INDICATE WRITE OPERATION
E020 B04A               2428            MOV     AL,04AH                 ; WILL WRITE TO THE DISKETTE
E022 E8A701             2429            CALL    DMA_SETUP               ; SET UP THE DMA
E025 B44D               2430            MOV     AH,04DH                 ; ESTABLISH THE FORMAT COMMAND
E027 EB24               2431            JMP     SHORT RW_OPN            ; DO THE OPERATION
E029                    2432    J10:                                    ; CONTINUATION OF RW_OPN FOR FMT
E029 BB0700             2433            MOV     BX,7                    ; GET THE
E02C E84101             2434            CALL    GET_PARM                ;  BYTES/SECTOR VALUE TO NEC
E02F BB0900             2435            MOV     BX,9                    ; GET THE
E032 E83B01             2436            CALL    GET_PARM                ;  SECTORS/TRACK VALUE TO NEC
E035 BB0F00             2437            MOV     BX,15                   ; GET THE
E038 E83501             2438            CALL    GET_PARM                ;  GAP LENGTH VALUE TO NEC
E03B BB1100             2439            MOV     BX,17                   ; GET THE FILLER BYTE
E03E E9AB00             2440            JMP     J16                     ;  TO THE CONTROLLER
                        2441    DISK_FORMAT     ENDP
                        2442
                        2443    ;------ DISKETTE WRITE ROUTINE
                        2444
E041                    2445    DISK_WRITE      PROC    NEAR
E041 800E3F0080     R   2446            OR      MOTOR_STATUS,80H        ; INDICATE WRITE OPERATION
E046 B04A               2447            MOV     AL,04AH                 ; DMA WRITE COMMAND
E048 E88101             2448            CALL    DMA_SETUP
E04B B445               2449            MOV     AH,045H                 ; NEC COMMAND TO WRITE TO DISKETTE
                        2450    DISK_WRITE      ENDP
                        2451    ;----- ALLOW WRITE ROUTINE TO FALL INTO RW_OPN
                        2452    ;--------------------------------------
                        2453    ; RW_OPN
                        2454    ;       THIS ROUTINE PERFORMS THE READ/WRITE/VERIFY OPERATION
                        2455    ;--------------------------------------
E04D                    2456    RW_OPN  PROC    NEAR
E04D 7308               2457            JNC     J11                     ; TEST FOR DMA ERROR
E04F C606410009     R   2458            MOV     DISKETTE_STATUS,DMA_BOUNDARY    ; SET ERROR
E054 B000               2459            MOV     AL,0                    ; NO SECTORS TRANSFERRED
E056 C3                 2460            RET                             ; RETURN TO MAIN ROUTINE
E057                    2461    J11:                                    ; DO_RW_OPN
E057 50                 2462            PUSH    AX              ; SAVE THE COMMAND
                        2463
                        2464    ;------ TURN ON THE MOTOR AND SELECT THE DRIVE
                        2465
E058 51                 2466            PUSH    CX              ; SAVE THE T/S PARMS
E059 8ACA               2467            MOV     CL,DL           ; GET DRIVE NUMBER AS SHIFT COUNT
E05B B001               2468            MOV     AL,1            ; MASK FOR DETERMINING MOTOR BIT
E05D D2E0               2469            SAL     AL,CL           ; SHIFT THE MASK BIT
E05F FA                 2470            CLI                     ; NO INTERRUPTS WHILE DETERMINING MOTOR STATUS
E060 C6064000FF     R   2471            MOV     MOTOR_COUNT,0FFH        ; SET LARGE COUNT DURING OPERATION
E065 84063F00       R   2472            TEST    AL,MOTOR_STATUS ; TEST THAT MOTOR FOR OPERATING
E069 7531               2473            JNZ     J14             ; IF RUNNING, SKIP THE WAIT
E06B 80263F00F0     R   2474            AND     MOTOR_STATUS,0F0H       ;-TURN OFF ALL MOTOR BITS
E070 08063F00       R   2475            OR      MOTOR_STATUS,AL         ; TURN ON THE CURRENT MOTOR
E074 FB                 2476            STI                     ; INTERRUPTS BACK ON
E075 B010               2477            MOV     AL,10H          ; MASK BIT
E077 D2E0               2478            SAL     AL,CL           ; DEVELOP BIT MASK FOR MOTOR ENABLE
E079 0AC2               2479            OR      AL,DL           ; GET DRIVE SELECT BITS IN
E07B 0C0C               2480            OR      AL,0CH          ; NO RESET, ENABLE DMA/INT
E07D 52                 2481            PUSH    DX              ; SAVE REG
E07E BAF203             2482            MOV     DX,03F2H        ; CONTROL PORT ADDRESS
E081 EE                 2483            OUT     DX,AL
```



```
LOC OBJ                 LINE    SOURCE

E082 5A                 2484            POP     DX              ; RECOVER REGISTERS
                        2485
                        2486    ;------ WAIT FOR MOTOR IF WRITE OPERATION
                        2487
E083 F6063F0080     R   2488            TEST    MOTOR_STATUS,80H        ; IS THIS A WRITE
E088 7412               2489            JZ      J14                     ; NO, CONTINUE WITHOUT WAIT
E08A BB1400             2490            MOV     BX,20                   ; GET THE MOTOR WAIT
E08D E8E000             2491            CALL    GET_PARM                ;  PARAMETER
E090 0AE4               2492            OR      AH,AH                   ; TEST FOR NO WAIT
E092                    2493    J12:                                    ; TEST_WAIT_TIME
E092 7408               2494            JZ      J14                     ; EXIT WITH TIME EXPIRED
E094 2BC9               2495            SUB     CX,CX                   ; SET UP 1/8 SECOND LOOP TIME
E096 E2FE               2496    J13:    LOOP    J13                     ; WAIT FOR THE REQUIRED TIME
E098 FECC               2497            DEC     AH                      ; DECREMENT TIME VALUE
E09A EBF6               2498            JMP     J12                     ; ARE WE DONE YET
                        2499
E09C                    2500    J14:                                    ; MOTOR_RUNNING
E09C FB                 2501            STI                     ; INTERRUPTS BACK ON FOR BYPASS WAIT
E09D 59                 2502            POP     CX
                        2503
                        2504    ;------ DO THE SEEK OPERATION
                        2505
E09E E8E000             2506            CALL    SEEK            ; MOVE TO CORRECT TRACK
E0A1 58                 2507            POP     AX              ; RECOVER COMMAND
E0A2 8AFC               2508            MOV     BH,AH           ; SAVE COMMAND IN BH
E0A4 B600               2509            MOV     DH,0            ; SET NO SECTORS READ IN CASE OF ERROR
E0A6 7248               2510            JC      J17             ; IF ERROR, THEN EXIT AFTER MOTOR OFF
E0A8 BEF3ED90       R   2511            MOV     SI,OFFSET J17   ; DUMMY RETURN ON STACK FOR NEC_OUTPUT
E0AC 56                 2512            PUSH    SI              ;  SO THAT IT WILL RETURN TO MOTOR OFF LOCATION
                        2513
                        2514    ;------ SEND OUT THE PARAMETERS TO THE CONTROLLER
                        2515
E0AD E89400             2516            CALL    NEC_OUTPUT      ; OUTPUT THE OPERATION COMMAND
E0B0 8A6601             2517            MOV     AH,[BP+1]       ; GET THE CURRENT HEAD NUMBER
E0B3 D0E4               2518            SAL     AH,1            ; MOVE IT TO BIT 2
E0B5 D0E4               2519            SAL     AH,1
E0B7 80E404             2520            AND     AH,4            ; ISOLATE THAT BIT
E0BA 0AE2               2521            OR      AH,DL           ; OR IN THE DRIVE NUMBER
E0BC E88500             2522            CALL    NEC_OUTPUT
                        2523
                        2524    ;------ TEST FOR FORMAT COMMAND
                        2525
E0BF 80FF4D             2526            CMP     BH,04DH         ; IS THIS A FORMAT OPERATION
E0C2 7503               2527            JNE     J15             ; NO, CONTINUE WITH R/W/V
E0C4 E962FF             2528            JMP     J10             ; IF SO, HANDLE SPECIAL
                        2529
E0C7 8AE5               2530    J15:    MOV     AH,CH           ; CYLINDER NUMBER
E0C9 E87800             2531            CALL    NEC_OUTPUT
E0CC 8A6601             2532            MOV     AH,[BP+1]       ; HEAD NUMBER FROM STACK
E0CF E87200             2533            CALL    NEC_OUTPUT
E0D2 8AE1               2534            MOV     AH,CL           ; SECTOR NUMBER
E0D4 E86D00             2535            CALL    NEC_OUTPUT
E0D7 BB0700             2536            MOV     BX,7            ; BYTES/SECTOR PARM FROM BLOCK
E0DA E89300             2537            CALL    GET_PARM        ;  TO THE NEC
E0DD BB0900             2538            MOV     BX,9            ; EOT PARM FROM BLOCK
E0E0 E88D00             2539            CALL    GET_PARM        ;  TO THE NEC
E0E3 BB0B00             2540            MOV     BX,11           ; GAP LENGTH PARM FROM BLOCK
E0E6 E88700             2541            CALL    GET_PARM        ;  TO THE NEC
E0E9 BB0D00             2542            MOV     BX,13           ; DTL PARM FROM BLOCK
E0EC                    2543    J16:                            ; RW_OPN_FINISH
E0EC E88100             2544            CALL    GET_PARM        ;  TO THE NEC
E0EF 5E                 2545            POP     SI              ; CAN NOW DISCARD THAT DUMMY RETURN ADDRESS
                        2546
                        2547    ;------ LET THE OPERATION HAPPEN
                        2548
E0F0 E8400I             2549            CALL    WAIT_INT        ; WAIT FOR THE INTERRUPT
E0F3                    2550    J17:                            ; MOTOR_OFF
E0F3 7245               2551            JC      J21             ; LOOK FOR ERROR
E0F5 E87301             2552            CALL    RESULTS         ; GET THE NEC STATUS
E0F8 723F               2553            JC      J20             ; LOOK FOR ERROR
                        2554
                        2555    ;------ CHECK THE RESULTS RETURNED BY THE CONTROLLER
                        2556
E0FA FC                 2557            CLD                     ; SET THE CORRECT DIRECTION
E0FB BE4200         R   2558            MOV     SI,OFFSET NEC_STATUS    ; POINT TO STATUS FIELD
E0FE AC                 2559            LODS    NEC_STATUS              ; GET ST0
E0FF 24C0               2560            AND     AL,0C0H                 ; TEST FOR NORMAL TERMINATION
```

```
LOC OBJ                 LINE    SOURCE

EE01 743B               2561            JZ      J22                     ; OPN_OK
EE03 3C40               2562            CMP     AL,040H                 ; TEST FOR ABNORMAL TERMINATION
EE05 7529               2563            JNZ     J18                     ; NOT ABNORMAL, BAD NEC
                        2564
                        2565    ;------ ABNORMAL TERMINATION, FIND OUT WHY
                        2566
EE07 AC                 2567            LODS    NEC_STATUS              ; GET ST1
EE08 D0E0               2568            SAL     AL,1                    ; TEST FOR EOT FOUND
EE0A B404               2569            MOV     AH,RECORD_NOT_FND
EE0C 7224               2570            JC      J19                     ; RW_FAIL
EE0E D0E0               2571            SAL     AL,1
EE10 D0E0               2572            SAL     AL,1                    ; TEST FOR CRC ERROR
EE12 B410               2573            MOV     AH,BAD_CRC
EE14 721C               2574            JC      J19                     ; RW_FAIL
EE16 D0E0               2575            SAL     AL,1                    ; TEST FOR DMA OVERRUN
EE18 B408               2576            MOV     AH,BAD_DMA
EE1A 7216               2577            JC      J19                     ; RW_FAIL
EE1C D0E0               2578            SAL     AL,1
EE1E D0E0               2579            SAL     AL,1                    ; TEST FOR RECORD NOT FOUND
EE20 B404               2580            MOV     AH,RECORD_NOT_FND
EE22 720E               2581            JC      J19                     ; RW_FAIL
EE24 D0E0               2582            SAL     AL,1
EE26 B403               2583            MOV     AH,WRITE_PROTECT        ; TEST FOR WRITE_PROTECT
EE28 7208               2584            JC      J19                     ; RW_FAIL
EE2A D0E0               2585            SAL     AL,1                    ; TEST MISSING ADDRESS MARK
EE2C B402               2586            MOV     AH,BAD_ADDR_MARK
EE2E 7202               2587            JC      J19                     ; RW_FAIL
                        2588
                        2589    ;------ NEC MUST HAVE FAILED
                        2590
EE30                    2591    J18:                                    ; RW-NEC-FAIL
EE30 B420               2592            MOV     AH,BAD_NEC
EE32                    2593    J19:                                    ; RW-FAIL
EE32 08264100     R     2594            OR      DISKETTE_STATUS,AH
EE36 E87701             2595            CALL    NUM_TRANS               ; HOW MANY WERE REALLT TRANSFERRED
                        2596                                            ; RW_ERR
EE39                    2597    J20:    RET                     ; RETURN TO CALLER
EE39 C3                 2598
                        2599    J21:                            ; RW_ERR_PES
EE3A                    2600            CALL    RESULTS         ; FLUSH THE RESULTS BUFFER
EE3A E82E01             2601            RET
EE3D C3                 2602
                        2603    ;------ OPERATION WAS SUCCESSFUL
                        2604
EE3E                    2605    J22:                                    ; OPN_OK
EE3E E86F01             2606            CALL    NUM_TRANS               ; HOW MANY GOT MOVED
EE41 32E4               2607            XOR     AH,AH                   ; NO ERRORS
EE43 C3                 2608            RET
                        2609    RW_OPN  ENDP
                        2610    ;--------------------------------------------
                        2611    ; NEC_OUTPUT
                        2612    ;       THIS ROUTINE SENDS A BYTE TO THE NEC CONTROLLER
                        2613    ;       AFTER TESTING FOR CORRECT DIRECTION AND CONTROLLER READY
                        2614    ;       THIS ROUTINE WILL TIME OUT IF THE BYTE IS NOT ACCEPTED
                        2615    ;       WITHIN A REASONABLE AMOUNT OF TIME, SETTING THE DISKETTE STATUS
                        2616    ;       ON COMPLETION
                        2617    ; INPUT
                        2618    ;       (AH)    BYTE TO BE OUTPUT
                        2619    ; OUTPUT
                        2620    ;       CY = 0  SUCCESS
                        2621    ;       CY = 1  FAILURE -- DISKETTE STATUS UPDATED
                        2622    ;               IF A FAILURE HAS OCCURRED, THE RETURN IS MADE ONE LEVEL
                        2623    ;               HIGHER THAN THE CALLER OF NEC_OUTPUT
                        2624    ;               THIS REMOVES THE REQUIREMENT OF TESTING AFTER EVERY CALL
                        2625    ;               OF NEC_OUTPUT
                        2626    ;       (AL) DESTROYED
                        2627    ;--------------------------------------------
EE44                    2628    NEC_OUTPUT      PROC    NEAR
EE44 52                 2629            PUSH    DX              ; SAVE REGISTERS
EE45 51                 2630            PUSH    CX
EE46 BAF403             2631            MOV     DX,03F4H        ; STATUS PORT
EE49 33C9               2632            XOR     CX,CX           ; COUNT FOR TIME OUT
EE4B                    2633    J23:
EE4B EC                 2634            IN      AL,DX           ; GET STATUS
EE4C A840               2635            TEST    AL,040H         ; TEST DIRECTION BIT
EE4E 740C               2636            JZ      J25             ; DIRECTION OK
EE50 E2F9               2637            LOOP    J23
EE52                    2638    J24:                            ; TIME_ERROR
```



```
LOC OBJ                 LINE    SOURCE

EE52 800E410080   R     2639            OR      DISKETTE_STATUS,TIME_OUT
EE57 59                 2640            POP     CX
EE58 5A                 2641            POP     DX              ; SET ERROR CODE AND RESTORE REGS
EE59 58                 2642            POP     AX              ; DISCARD THE RETURN ADDRESS
EE5A F9                 2643            STC                     ; INDICATE ERROR TO CALLER
EE5B C3                 2644            RET
                        2645
EE5C                    2646    J25:
EE5C 33C9               2647            XOR     CX,CX           ; RESET THE COUNT
EE5E                    2648    J26:
EE5E EC                 2649            IN      AL,DX           ; GET THE STATUS
EE5F A880               2650            TEST    AL,080H         ; IS IT READY
EE61 7504               2651            JNZ     J27             ; YES, GO OUTPUT
EE63 E2F9               2652            LOOP    J26             ; COUNT DOWN AND TRY AGAIN
EE65 EBEB               2653            JMP     J24             ; ERROR CONDITION
EE67                    2654    J27:                            ; OUTPUT
EE67 8AC4               2655            MOV     AL,AH           ; GET BYTE TO OUTPUT
EE69 BAF503             2656            MOV     DX,03F5H        ; DATA PORT
EE6C EE                 2657            OUT     DX,AL           ; OUTPUT THE BYTE
EE6D 59                 2658            POP     CX              ; RECOVER REGISTERS
EE6E 5A                 2659            POP     DX
EE6F C3                 2660            RET                     ; CY = 0 FROM TEST INSTRUCTION
                        2661    NEC_OUTPUT      ENDP
                        2662    ;--------------------------------------------
                        2663    ; GET_PARM
                        2664    ;  THIS ROUTINE FETCHES THE INDEXED POINTER FROM
                        2665    ;  THE DISK_BASE BLOCK POINTED AT BY THE DATA
                        2666    ;  VARIABLE DISK_POINTER
                        2667    ; A BYTE FROM THAT TABLE IS THEN MOVED INTO AH,
                        2668    ;  THE INDEX OF THAT BYTE BEING THE PARM IN BX
                        2669    ; ENTRY --
                        2670    ;   BX = INDEX OF BYTE TO BE FETCHED * 2
                        2671    ;        IF THE LOW BIT OF BX IS ON, THE BYTE IS IMMEDIATELY
                        2672    ;        OUTPUT TO THE NEC CONTROLLER
                        2673    ; EXIT --
                        2674    ;   AH = THAT BYTE FROM BLOCK
                        2675    ;--------------------------------------------
EE70                    2676    GET_PARM        PROC    NEAR
EE70 1E                 2677            PUSH    DS              ; SAVE SEGMENT
EE71 28C0               2678            SUB     AX,AX           ; ZERO TO AX
EE73 8ED8               2679            MOV     DS,AX
                        2680            ASSUME  DS:ABS0
EE75 C5367800           2681            LDS     SI,DISK_POINTER ; POINT TO BLOCK
EE79 D1EB               2682            SHR     BX,1            ; DIVIDE BX BY 2, AND SET FLAG FOR EXIT
EE7B 8A20               2683            MOV     AH,[SI+BX]      ; GET THE WORD
EE7D 1F                 2684            POP     DS              ; RESTORE SEGMENT
                        2685            ASSUME  DS:DATA
EE7E 72C4               2686            JC      NEC_OUTPUT      ; IF FLAG SET, OUTPUT TO CONTROLLER
EE80 C3                 2687            RET                     ; RETURN TO CALLER
                        2688    GET_PARM        ENDP
                        2689    ;--------------------------------------------
                        2690    ; SEEK
                        2691    ;       THIS ROUTINE WILL MOVE THE HEAD ON THE NAMED DRIVE
                        2692    ;       TO THE NAMED TRACK.  IF THE DRIVE HAS NOT BEEN ACCESSED
                        2693    ;       SINCE THE DRIVE RESET COMMAND HAS ISSUED, THE DRIVE WILL BE
                        2694    ;       RECALIBRATED.
                        2695    ; INPUT
                        2696    ;       (DL) = DRIVE TO SEEK ON
                        2697    ;       (CH) = TRACK TO SEEK TO
                        2698    ; OUTPUT
                        2699    ;       CY = 0 SUCCESS
                        2700    ;       CY = 1 FAILURE -- DISKETTE_STATUS SET ACCORDINGLY
                        2701    ;       (AX) DESTROYED
                        2702    ;--------------------------------------------
EE81                    2703    SEEK    PROC    NEAR
EE81 B001               2704            MOV     AL,1            ; ESTABLISH MASK FOR RECAL TEST
EE83 51                 2705            PUSH    CX              ; SAVE INPUT VALUES
EE84 8ACA               2706            MOV     CL,DL           ; GET DRIVE VALUE INTO CL
EE86 D2C0               2707            ROL     AL,CL           ; SHIFT IT BY THE DRIVE VALUE
EE88 59                 2708            POP     CX              ; RECOVER TRACK VALUE
EE89 84063E00     R     2709            TEST    AL,SEEK_STATUS  ; TEST FOR RECAL REQUIRED
EE8D 7513               2710            JNZ     J28             ; NO_RECAL
EE8F 08063E00     R     2711            OR      SEEK_STATUS,AL  ; TURN ON THE NO RECAL BIT IN FLAG
EE93 B407               2712            MOV     AH,07H          ; RECALIBRATE COMMAND
EE95 E8ACFF             2713            CALL    NEC_OUTPUT
EE98 8AE2               2714            MOV     AH,DL
EE9A E8A7FF             2715            CALL    NEC_OUTPUT      ; OUTPUT THE DRIVE NUMBER
```

```
LOC OBJ                 LINE    SOURCE


EE90 E87200             2716            CALL    CHK_STAT_2      ; GET THE INTERUPT AND SENSE INT STATUS
EEA0 7229               2717            JC      J32             ; SEEK_ERROR
                        2718
                        2719    ;----- DRIVE IS IN SYNCH WITH CONTROLLER, SEEK TO TRACK
                        2720
EEA2                    2721    J28:
EEA2 B40F               2722            MOV     AH,0FH          ; SEEK COMMAND TO NEC
EEA4 E870FF             2723            CALL    NEC_OUTPUT
EEA7 8AE2               2724            MOV     AH,DL           ; DRIVE NUMBER
EEA9 E868FF             2725            CALL    NEC_OUTPUT
EEAC 8AE5               2726            MOV     AH,CH           ; TRACK NUMBER
EEAE E863FF             2727            CALL    NEC_OUTPUT
EEB1 E85E00             2728            CALL    CHK_STAT_2      ; GET ENDING INTERRUPT AND SENSE STATUS
                        2729
                        2730    ;----- WAIT FOR HEAD SETTLE
                        2731
EEB4 9C                 2732            PUSHF                   ; SAVE STATUS FLAGS
EEB5 BB1200             2733            MOV     BX,18           ; GET HEAD SETTLE PARAMETER
EEB8 E8B5FF             2734            CALL    GET_PARM
EEBB 51                 2735            PUSH    CX              ; SAVE REGISTER
EEBC                    2736    J29:                            ; HEAD_SETTLE
EEBC B92602             2737            MOV     CX,550          ; 1 MS LOOP
EEBF 0AE4               2738            OR      AH,AH           ; TEST FOR TIME EXPIRED
EEC1 7406               2739            JZ      J31
EEC3 E2FE               2740    J30:    LOOP    J30             ; DELAY FOR 1 MS
EEC5 FECC               2741            DEC     AH              ; DECREMENT THE COUNT
EEC7 EBF3               2742            JMP     J29             ; DO IT SOME MORE
EEC9                    2743    J31:
EEC9 59                 2744            POP     CX              ; RECOVER STATE
EECA 9D                 2745            POPF
EECB                    2746    J32:                            ; SEEK_ERROR
EECB C3                 2747            RET                     ; RETURN TO CALLER
                        2748    SEEK    ENDP
                        2749    ;--------------------------------------------
                        2750    ; DMA_SETUP
                        2751    ;       THIS ROUTINE SETS UP THE DMA FOR READ/WRITE/VERIFT
                        2752    ;       OPERATIONS.
                        2753    ; INPUT
                        2754    ;       (AL) = MODE BYTE FOR THE DMA
                        2755    ;       (ES:BX) - ADDRESS TO READ/WRITE THE DATA
                        2756    ; OUTPUT
                        2757    ;       (AX) DESTROYED
                        2758    ;--------------------------------------------
EECC                    2759    DMA_SETUP       PROC    NEAR
EECC 51                 2760            PUSH    CX              ; SAVE THE REGISTER
EECD E60C               2761            OUT     DMA+12,AL       ; SET THE FIRST/LAST F/F
EECF E60B               2762            OUT     DMA+11,AL       ; OUTPUT THE MODE BYTE
EED1 8CC0               2763            MOV     AX,ES           ; GET THE ES VALUE
EED3 B104               2764            MOV     CL,4            ; SHIFT COUNT
EED5 D3C0               2765            ROL     AX,CL           ; ROTATE LEFT
EED7 8AE8               2766            MOV     CH,AL           ; GET HIGHEST NIBLE OF ES TO CH
EED9 24F0               2767            AND     AL,0F0H         ; ZERO THE LOW NIBBLE FROM SEGMENT
EEDB 03C3               2768            ADD     AX,BX           ; TEST FOR CARRT FROM ADDITION
EEDD 7302               2769            JNC     J33
EEDF FEC5               2770            INC     CH              ; CARRY MEANS HIGH 4 BITS MUST BE INC
EEE1                    2771    J33:
EEE1 50                 2772            PUSH    AX              ; SAVE START ADDRESS
EEE2 E604               2773            OUT     DMA+4,AL        ; OUTPUT LOW ADDRESS
EEE4 8AC4               2774            MOV     AL,AH
EEE6 E604               2775            OUT     DMA+4,AL        ; OUTPUT HIGH ADDRESS
EEE8 8AC5               2776            MOV     AL,CH           ; GET HIGH 4 BITS
EEEA 240F               2777            AND     AL,0FH
EEEC E681               2778            OUT     081H,AL         ; OUTPUT THE HIGH 4 BITS TO PAGE REGISTER
                        2779
                        2780    ;------ DETERMINE COUNT
                        2781
EEEE 8AE6               2782            MOV     AH,DH           ; NUMBER OF SECTORS
EEF0 2AC0               2783            SUB     AL,AL           ;  TIMES 256 INTO AX
EEF2 D1E8               2784            SHR     AX,1            ; SECTORS * 128 INTO AX
EEF4 50                 2785            PUSH    AX
EEF5 BB0600             2786            MOV     BX,6            ; GET THE BYTES/SECTOR PARM
EEF8 E875FF             2787            CALL    GET_PARM
EEFB 8ACC               2788            MOV     CL,AH           ; USE AS SHIFT COUNT (0=128, 1=256 ETC)
EEFD 58                 2789            POP     AX
EEFE D3E0               2790            SHL     AX,CL           ; MULTIPLY BY CORRECT AMOUNT
EF00 48                 2791            DEC     AX              ; -1 FOR DMA VALUE
EF01 50                 2792            PUSH    AX              ; SAVE COUNT VALUE
```



```
LOC OBJ                 LINE    SOURCE

EF02 E605               2793            OUT     DMA+5,AL        ; LOW BYTE OF COUNT
EF04 8AC4               2794            MOV     AL,AH
EF06 E605               2795            OUT     DMA+5,AL        ; HIGH BYTE OF COUNT
EF08 59                 2796            POP     CX              ; RECOVER COUNT VALUE
EF09 58                 2797            POP     AX              ; RECOVER ADDRESS VALUE
EF0A 03C1               2798            ADD     AX,CX           ; ADD, TEST FOR 64K OVERFLOW
EF0C 59                 2799            POP     CX              ; RECOVER REGISTER
EF0D B002               2800            MOV     AL,2            ; MODE FOR 8237
EF0F E60A               2801            OUT     DMA+10,AL       ; INITIALIZE THE DISKETTE CHANNEL
EF11 C3                 2802            RET                     ; RETURN TO CALLER, CFL SET BY ABOVE IF ERROR
                        2803    DMA_SETUP       ENDP
                        2804    ;--------------------------------------------
                        2805    ; CHK_STAT_2
                        2806    ;       THIS ROUTINE HANDLES THE INTERRUPT RECEIVED AFTER
                        2807    ;       A RECALIBRATE, SEEK, OR RESET TO THE ADAPTER.
                        2808    ;       THE INTERRUPT IS WAITED FOR, THE INTERRUPT STATUS SENSED,
                        2809    ;       AND THE RESULT RETURNED TO THE CALLER.
                        2810    ; INPUT
                        2811    ;       NONE
                        2812    ; OUTPUT
                        2813    ;       CY = 0 SUCCESS
                        2814    ;       CY = 1 FAILURE -- ERROR IS IN DISKETTE_STATUS
                        2815    ;       (AX) DESTROYED
                        2816    ;--------------------------------------------
EF12                    2817    CHK_STAT_2      PROC    NEAR
EF12 E81E00             2818            CALL    WAIT_INT        ; WAIT FOR THE INTERRUPT
EF15 7214               2819            JC      J34             ; IF ERROR, RETURN IT
EF17 B408               2820            MOV     AH,08H          ; SENSE INTERRUPT STATUS COMMAND
EF19 E828FF             2821            CALL    NEC_OUTPUT
EF1C E84C00             2822            CALL    RESULTS         ; READ IN THE RESULTS
EF1F 720A               2823            JC      J34             ; CHK2_RETURN
EF21 A04200         R   2824            MOV     AL,NEC_STATUS   ; GET THE FIRST STATUS BYTE
EF24 2460               2825            AND     AL,060H         ; ISOLATE THE BITS
EF26 3C60               2826            CMP     AL,060H         ; TEST FOR CORRECT VALUE
EF28 7402               2827            JZ      J35             ; IF ERROR, GO MARK IT
EF2A F8                 2828            CLC                     ; GOOD RETURN
EF2B                    2829    J34:
EF2B C3                 2830            RET                     ; RETURN TO CALLER
EF2C                    2831    J35:                            ; CHK2_ERROR
EF2C 800E410040     R   2832            OR      DISKETTE_STATUS,BAD_SEEK
EF31 F9                 2833            STC                     ; ERROR RETURN CODE
EF32 C3                 2834            RET
                        2835    CHK_STAT_2      ENDP
                        2836    ;--------------------------------------------
                        2837    ; WAIT_INT
                        2838    ;       THIS ROUTINE WAITS FOR AN INTERRUPT TO OCCUR
                        2839    ;       A TIME OUT ROUTINE TAKES PLACE DURING THE WAIT, SO
                        2840    ;       THAT AN ERROR MAY BE RETURNED IF THE DRIVE IS NOT READY
                        2841    ; INPUT
                        2842    ;       NONE
                        2843    ; OUTPUT
                        2844    ;       CY = 0 SUCCESS
                        2845    ;       CY = 1 FAILURE -- DISKETTE_STATUS IS SET ACCORDINGLY
                        2846    ;       (AX) DESTROYED
                        2847    ;--------------------------------------------
EF33                    2848    WAIT_INT        PROC    NEAR
EF33 FB                 2849            STI             ; TURN ON INTERRUPTS, JUST IN CASE
EF34 53                 2850            PUSH    BX
EF35 51                 2851            PUSH    CX              ; SAVE REGISTERS
EF36 B302               2852            MOV     BL,2            ; CLEAR THE COUNTERS
EF38 33C9               2853            XOR     CX,CX           ; FOR 2 SECOND WAIT
EF3A                    2854    J36:
EF3A F6063E0080     R   2855            TEST    SEEK_STATUS,INT_FLAG    ; TEST FOR INTERRUPT OCCURRING
EF3F 750C               2856            JNZ     J37
EF41 E2F7               2857            LOOP    J36             ; COUNT DOWN WHILE WAITING
EF43 FECB               2858            DEC     BL              ; SECOND LEVEL COUNTER
EF45 75F3               2859            JNZ     J36
EF47 800E410080     R   2860            OR      DISKETTE_STATUS,TIME_OUT        ; NOTHING HAPPENED
EF4C F9                 2861            STC                     ; ERROR RETURN
EF4D                    2862    J37:
EF4D 9C                 2863            PUSHF                   ; SAVE CURRENT CARRT
EF4E 80263E007F     R   2864            AND     SEEK_STATUS,NOT INT_FLAG        ; TURN OFF INTERRUPT FLAG
EF53 9D                 2865            POPF                    ; RECOVER CARRT
EF54 59                 2866            POP     CX
EF55 5B                 2867            POP     BX              ; RECOVER REGISTERS
EF56 C3                 2868            RET                     ; GOOD RETURN CODE COMES FROM TEST INST
                        2869    WAIT_INT        ENDP
```

```
LOC OBJ                 LINE    SOURCE

                        2870    ;--------------------------------------------
                        2871    ; DISK_INT
                        2872    ;       THIS ROUTINE HANDLES THE DISKETTE INTERRUPT
                        2873    ;INPUT
                        2874    ;       NONE
                        2875    ; OUTPUT
                        2876    ;       THE INTERRUPT FLAG IS SET IS SEEK_STATUS
                        2877    ;--------------------------------------------
EF57                    2878    DISK_INT        PROC    FAR
EF57 FB                 2879            STI                     ; RE ENABLE INTERRUPTS
EF58 1E                 2880            PUSH    DS
EF59 50                 2881            PUSH    AX
EF5A B84000         R   2882            MOV     AX,DATA
EF5D 8ED8               2883            MOV     DS,AX
EF5F 800E3E0080     R   2884            OR      SEEK_STATUS.INT_FLAG
EF64 B020               2885            MOV     AL,20H                  ; END OF INTERRUPT MARKER
EF66 E620               2886            OUT     20H,AL                  ; INTERRUPT CONTROL PORT
EF68 58                 2887            POP     AX
EF69 1F                 2888            POP     DS                      ; RECOVER SYSTEM
EF6A CF                 2889            IRET                            ; RETURN FROM INTERRUPT
                        2890    DISK_INT        ENDP
                        2891    ;--------------------------------------------
                        2892    ; RESULTS
                        2893    ;       THIS ROUTINE WILL READ ANYTHING THAT THE NEC CONTROLLER
                        2894    ;       HAS TO SAY FOLLOWING AN INTERRUPT.
                        2895    ; INPUT
                        2896    ;       NONE
                        2897    ; OUTPUT
                        2898    ;       CY = 0  SUCCESSFUL TRANSFER
                        2899    ;       CY = 1  FAILURE -- TIME OUT IN WAITING FOR STATUS
                        2900    ;       NEC_STATUS AREA HAS STATUS BYTE LOADED INTO IT
                        2901    ;       (AH) DESTROYED
                        2902    ;--------------------------------------------
EF6B                    2903    RESULTS PROC    NEAR
EF6B FC                 2904            CLD
EF6C BF4200         R   2905            MOV     DI,OFFSET NEC_STATUS    ; POINTER TO DATA AREA
EF6F 51                 2906            PUSH    CX                      ; SAVE COUNTER
EF70 52                 2907            PUSH    DX
EF71 53                 2908            PUSH    BX
EF72 B307               2909            MOV     BL,7                    ; MAX STATUS BYTES
                        2910
                        2911    ;------ WAIT FOR REQUEST FOR MASTER
                        2912
EF74                    2913    J38:                                    ; INPUT_LOOP
EF74 33C9               2914            XOR     CX,CX                   ; COUNTER
EF76 BAF403             2915            MOV     DX,03F4H                ; STATUS PORT
EF79                    2916    J39:                                    ; WAIT FOR MASTER
EF79 EC                 2917            IN      AL,DX                   ; GET STATUS
EF7A A880               2918            TEST    AL,080H                 ; MASTER READY
EF7C 750C               2919            JNZ     J40A                    ; TEST_DIR
EF7E E2F9               2920            LOOP    J39                     ; WAIT_MASTER
EF80 800E410080     R   2921            OR      DISKETTE_STATUS,TIME_OUT
EF65                    2922    J40:                                    ; RESULTS_ERROR
EF85 F9                 2923            STC                             ; SET ERROR RETURN
EF86 5B                 2924            POP     BX
EF87 5A                 2925            POP     DX
EF88 59                 2926            POP     CX
EF89 C3                 2927            RET
                        2928
                        2929    ;------ TEST THE DIRECTION BIT
                        2930
EF8A EC                 2931    J40A:   IN      AL,DX           ; GET STATUS REG AGAIN
EF8B A840               2932            TEST    AL,040H         ; TEST DIRECTION BIT
EF8D 7507               2933            JNZ     J42             ; OK TO READ STATUS
EF8F                    2934    J41:                            ; NEC_FAIL
EF8F 800E410020     R   2935            OR      DISKETTE_STATUS,BAD_NEC
EF94 EBEF               2936            JMP     J40             ; RESULTS_ERROR
                        2937
                        2938    ;------ READ IN THE STATUS
                        2939
EF96                    2940    J42:                            ; INPUT_STAT
EF96 42                 2941            INC     DX              ; POINT AT DATA PORT
EF97 EC                 2942            IN      AL,DX           ; GET THE DATA
EF98 8805               2943            MOV     [DI],AL         ; STORE THE BYTE
EF9A 47                 2944            INC     DI              ; INCREMENT THE POINTER
EF9B B90A00             2945            MOV     CX,10           ; LOOP TO KILL TIME FOR NEC
```



```
LOC OBJ                 LINE    SOURCE

EF9E E2FE               2946    J43:    LOOP    J43
EFA0 4A                 2947            DEC     DX              ; POINT AT STATUS PORT
EFA1 EC                 2948            IN      AL,DX           ; GET STATUS
EFA2 A810               2949            TEST    AL,010H         ; TEST FOR NEC STILL BUSY
EFA4 7406               2950            JZ      J44             ; RESULTS DONE
EFA6 FECB               2951            DEC     BL              ; DECREMENT THE STATUS COUNTER
EFA8 75CA               2952            JNZ     J38             ; GO BACK FOR MORE
EFAA EBE3               2953            JMP     J41             ; CHIP HAS FAILED
                        2954
                        2955    ;------ RESULT OPERATION IS DONE
                        2956
EFAC                    2957    J44:
EFAC 5B                 2958            POP     BX
EFAD 5A                 2959            POP     DX
EFAE 59                 2960            POP     CX              ; RECOVER REGISTERS
EFAF C3                 2961            RET                     ; GOOD RETURN CODE FROM TEST INST
                        2962    ;--------------------------------------------
                        2963    ; NUM_TRANS
                        2964    ;       THIS ROUTINE CALCULATES THE NUMBER OF SECTORS THAT
                        2965    ;       WERE ACTUALLY TRANSFERRED TO/FROM THE DISKETTE
                        2966    ; INPUT
                        2967    ;       (CH) = CYLINDER OF OPERATION
                        2968    ;       (CL) = START SECTOR OF OPERATION
                        2969    ; OUTPUT
                        2970    ;       (AL) = NUMBER ACTUALLY TRANSFERRED
                        2971    ;       NO OTHER REGISTERS MODIFIED
                        2972    ;--------------------------------------------
EFB0                    2973    NUM_TRANS       PROC    NEAR
EFB0 A04500         R   2974            MOV     AL,NEC_STATUS+3         ; GET CYLINDER ENDED UP ON
EFB3 3AC5               2975            CMP     AL,CH                   ; SAME AS WE STARTED
EFB5 A04700         R   2976            MOV     AL,NEC_STATUS+5         ; GET ENDING SECTOR
EFB8 740A               2977            JZ      J45                     ; IF ON SAME CTL, THEN NO ADJUST
EFBA BB0800             2978            MOV     BX,8
EFBD E8B0FE             2979            CALL    GET_PARM                ; GET EOT VALUE
EFC0 8AC4               2980            MOV     AL,AH                   ;  INTO AL
EFC2 FEC0               2981            INC     AL                      ; USE EOT+1 FOR CALCULATION
EFC4 2AC1               2982    J45:    SUB     AL,CL                   ; SUBTRACT START FROM END
EFC6 C3                 2983            RET
                        2984    NUM_TRANS       ENDP
                        2985    RESULTS ENDP
                        2986
                        2987    ;--------------------------------------------
                        2988    ; DISK_BASE
                        2989    ;  THIS IS THE SET OF PARAMETERS REQUIRED FOR
                        2990    ;  DISKETTE OPERATION.  THEY ARE POINTED AT BY THE
                        2991    ;  DATA VARIABLE DISK_POINTER.  TO MODIFY THE PARAMETERS,
                        2992    ;  BUILD ANOTHER PARAMETER BLOCK AND POINT AT IT
                        2993    ;--------------------------------------------
                        2994
EFC7                    2995    DISK_BASE       LABEL   BYTE
EFC7 CF                 2996            DB      11001111B       ; SRT=C, HD UNLOAD=0F - 1ST SPECIFY BYTE
EFC8 02                 2997            DB      2               ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
EFC9 25                 2998            DB      MOTOR_WAIT      ; WAIT AFTER OPN TIL MOTOR OFF
EFCA 02                 2999            DB      2               ; 512 BYTES/SECTOR
EFCB 0B                 3000            DB      8               ; EOT ( LAST SECTOR ON TRACK)
EFCC 2A                 3001            DB      02AH            ; GAP LENGTH
EFCD FF                 3002            DB      0FFH            ; DTL
EFCE 50                 3003            DB      050H            ; GAP LENGTH FOR FORMAT
EFCF F6                 3004            DB      0F6H            ; FILL BYTE FOR FORMAT
EFD0 19                 3005            DB      25              ; HEAD SETTLE TIME (MILLISECONDS)
EFD1 04                 3006            DB      4               ; MOTOR START TIME (1/8 SECONDS)
```

```
LOC OBJ                 LINE    SOURCE

                        3007    ;--- INT 17 --------------------------------
                        3008    ;PRINTER_IO
                        3009    ;       THIS ROUTINE PROVIDES COMMUNICATION WITH THE PRINTER
                        3010    ;       (AH)=0  PRINT THE CHARACTER IN (AL)
                        3011    ;               ON RETURN, AH=1 IF CHARACTER COULD NOT BE PRINTED (TIME OUT)
                        3012    ;               OTHER BITS SET AS ON NORMAL STATUS CALL
                        3013    ;       (AH)=1  INITIALIZE THE PRINTER PORT
                        3014    ;               RETURNS WITH (AH) SET WITH PRINTER STATUS
                        3015    ;       (AH)=2  READ THE PRINTER STATUS INTO (AH)
                        3016    ;               7       6       5       4       3       2-1     0
                        3017    ;               |       |       |       |       |       |       |_ TIME OUT
                        3018    ;               |       |       |       |       |       |_ UNUSED
                        3019    ;               |       |       |       |       |_ 1 = I/O ERROR
                        3020    ;               |       |       |       |_ 1 = SELECTED
                        3021    ;               |       |       |_ 1 = OUT OF PAPER
                        3022    ;               |       |_ 1 = ACKNOWLEDGE
                        3023    ;               |_ 1 = BUSY
                        3024    ;
                        3025    ;       (DX) = PRINTER TO BE USED (0,1,2) CORRESPONDING TO ACTUAL VALUES
                        3026    ;               IN PRINTER_BASE AREA
                        3027    ; DATA AREA PRINTER_BASE CONTAINS THE BASE ADDRESS OF THE PRINTER CARD(S)
                        3028    ;   AVAILABLE (LOCATED AT BEGINNING OF DATA SEGMENT, 408H ABSOLUTE, 3 WORDS)
                        3029    ;REGISTERS      AH IS MODIFIED
                        3030    ;               ALL OTHERS UNCHANGED
                        3031    ;-----------------------------------------------
                        3032            ASSUME  CS:CODE,DS:DATA
EFD2                    3033    PRINTER_IO      PROC    FAR
EFD2 FB                 3034            STI                     ; INTERRUPTS BACK ON
EFD3 1E                 3035            PUSH    DS              ; SAVE SEGMENT
EFD4 52                 3036            PUSH    DX
EFD5 56                 3037            PUSH    SI
EFD6 51                 3038            PUSH    CX
EFD7 53                 3039            PUSH    BX
EFD8 BE4000         R   3040            MOV     SI,DATA
EFDB 8EDE               3041            MOV     DS,SI           ; ESTABLISH PRINTER SEGMENT
EFDD 8BF2               3042            MOV     SI,DX           ; GET PRINTER PARM
EFDF D1E6               3043            SHL     SI,1            ; WORD OFFSET INTO TABLE
EFE1 8B940800       R   3044            MOV     DX,PRINTER_BASE[SI]     ; GET BASE ADDRESS FOR PRINTER CARD
EFE5 0BD2               3045            OR      DX,DX           ; TEST DX FOR ZERO, INDICATING NO PRINTER
EFE7 740C               3046            JZ      B1              ; RETURN
EFE9 0AE4               3047            OR      AH,AH           ; TEST FOR (AH)=0
EFEB 740E               3048            JZ      B2              ; PRINT_AL
EFED FECC               3049            DEC     AH              ; TEST FOR (AH)=1
EFEF 7442               3050            JZ      B8              ; INIT_PRT
EFF1 FECC               3051            DEC     AH              ; TEST FOR (AH)=2
EFF3 742A               3052            JZ      B5              ; PRINTER STATUS
EFF5                    3053    B1:                             ; RETURN
EFF5 5B                 3054            POP     BX
EFF6 59                 3055            POP     CX
EFF7 5E                 3056            POP     SI              ; RECOVER REGISTERS
EFF8 5A                 3057            POP     DX              ; RECOVER REGISTERS
EFF9 1F                 3058            POP     DS
EFFA CF                 3059            IRET
                        3060
                        3061    ;------ PRINT THE CHARACTER IN (AL)
                        3062
EFFB                    3063    B2:
EFFB 50                 3064            PUSH    AX              ; SAVE VALUE TO PRINT
EFFC B30A               3065            MOV     BL,10           ; TIME OUT VALUE
EFFE 33C9               3066            XOR     CX,CX           ; ESTABLISH SHIFT COUNT
F000 EE                 3067            OUT     DX,AL           ; OUTPUT CHAR TO PORT
F001 42                 3068            INC     DX              ; POINT TO STATUS PORT
F002                    3069    B3:                             ; WAIT_BUSY
F002 EC                 3070            IN      AL,DX           ; GET STATUS
F003 8AE0               3071            MOV     AH,AL           ; STATUS TO AH ALSO
F005 A880               3072            TEST    AL,80H          ; IS THE PRINTER CURRENTLY BUST
F007 750E               3073            JNZ     B4              ; OUT_STROBE
F009 E2F7               3074            LOOP    B3              ; DECREMENT COUNT ON TIME OUT
F00B FECB               3075            DEC     BL
F00D 75F3               3076            JNZ     B3              ; WAIT FOR NOT BUSY
F00F 80CC01             3077            OR      AH,1            ; SET ERROR FLAG
F012 80E4F9             3078            AND     AH,0F9H         ; TURN OFF THE OTHER BITS
F015 EB14               3079            JMP     SHORT B7        ; RETURN WITH ERROR FLAG SET
F017                    3080    B4:                             ; OUT_STROBE
F017 B00D               3081            MOV     AL,00H          ; SET THE STROBE HIGH
F019 42                 3082            INC     DX              ; STROBE IS BIT 0 OF PORT C OF 8255
```



```
LOC OBJ                 LINE    SOURCE

F01A EE                 3083            OUT     DX,AL
F01B B00C               3084            MOV     AL,0CH          ; SET THE STROBE LOW
F01D EE                 3085            OUT     DX,AL
F01E 58                 3086            POP     AX              ; RECOVER THE OUTPUT CHAR
                        3087
                        3088    ;------ PRINTER STATUS
                        3089
F01F                    3090    B5:
F01F 50                 3091            PUSH    AX              ; SAVE AL REG
F020                    3092    B6:
F020 8B940800       R   3093            MOV     DX,PRINTER_BASE[SI]
F024 42                 3094            INC     DX
F025 EC                 3095            IN      AL,DX           ; GET PRINTER STATUS
F026 8AE0               3096            MOV     AH,AL
F028 80E4F8             3097            AND     AH,0F8H         ; TURN OFF UNUSED BITS
F02B                    3098    B7:                             ; STATUS_SET
F02B 5A                 3099            POP     DX              ; RECOVER AL REG
F02C 8AC2               3100            MOV     AL,DL           ; GET CHARACTER INTO AL
F02E 80F448             3101            XOR     AH,48H          ; FLIP A COUPLE OF BITS
F031 EBC2               3102            JMP     B1              ; RETURN FROM ROUTINE
                        3103
                        3104    ;------ INITIALIZE THE PRINTER PORT
                        3105
F033                    3106    B8:
F033 50                 3107            PUSH    AX              ; SAVE AL
F034 83C202             3108            ADD     DX,2            ; POINT TO OUTPUT PORT
F037 B008               3109            MOV     AL,8            ; SET INIT LINE LOW
F039 EE                 3110            OUT     DX,AL
F03A B8E803             3111            MOV     AX,1000
F03D                    3112    B9:                             ; INIT_LOOP
F03D 48                 3113            DEC     AX              ; LOOP FOR RESET TO TAKE
F03E 75FD               3114            JNZ     B9              ; INIT_LOOP
F040 B00C               3115            MOV     AL,0CH          ; NO INTERRUPTS, NON AUTO LF, INIT HIGH
F042 EE                 3116            OUT     DX,AL
F043 EB0B               3117            JMP     B6              ; PRT_STATUS_1
                        3118    PRINTER_IO      ENDP
                        3119    ;--- INT 10 --------------------------------
                        3120    ; VIDEO_IO
                        3121    ;       THESE ROUTINES PROVIDE THE CRT INTERFACE
                        3122    ;       THE FOLLOWING FUNCTIONS ARE PROVIDED:
                        3123    ;       (AH)=0  SET MODE (AL) CONTAINS MODE VALUE
                        3124    ;               (AL)=0 40X25 BW (POWER ON DEFAULT)
                        3125    ;               (AL)=1 40X25 COLOR
                        3126    ;               (AL)=2  80X25 BW
                        3127    ;               (AL)=3  80X25 COLOR
                        3128    ;               GRAPHICS MODES
                        3129    ;               (AL)=4  320X200 COLOR
                        3130    ;               (AL)=5  320X200 BW
                        3131    ;               (AL)=6  640X200 BW
                        3132    ;               CRT MODE = 7 80X25 B&W CARD (USED INTERNAL TO VIDEO ONLY)
                        3133    ;               *** NOTE BW MODES OPERATE SAME AS COLOR MODES, BUT COLOR
                        3134    ;                       BURST IS NOT ENABLED
                        3135    ;       (AH)=1  SET CURSOR TYPE
                        3136    ;               (CH) =  BITS 4-0 = START LINE FOR CURSOR
                        3137    ;                       ** HARDWARE WILL ALWAYS CAUSE BLINK
                        3138    ;                       ** SETTING BIT 5 OR 6 WILL CAUSE ERRATIC BLINKING
                        3139    ;                          OR NO CURSOR AT ALL
                        3140    ;               (CL) =  BITS 4-0 = END LINE FOR CURSOR
                        3141    ;       (AH)=2  SET CURSOR POSITION
                        3142    ;               (DH,DL) = ROW,COLUMN  (0,0) IS UPPER LEFT
                        3143    ;               (BH) = PAGE NUMBER (MUST BE 0 FOR GRAPHICS MODES)
                        3144    ;       (AH)=3  READ CURSOR POSITION
                        3145    ;               (BH) = PAGE NUMBER (MUST BE 0 FOR GRAPHICS MODES)
                        3146    ;               ON EXIT (DH,DL) = ROW,COLUMN OF CURRENT CURSOR
                        3147    ;                       (CH,CL) = CURSOR MODE CURRENTLY SET
                        3148    ;       (AH)=4  READ LIGHT PEN POSITION
                        3149    ;               ON EXIT:
                        3150    ;               (AH) = 0 -- LIGHT PEN SWITCH NOT DOWN/NOT TRIGGERED
                        3151    ;               (AH) = 1 -- VALID LIGHT PEN VALUE IN REGISTERS
                        3152    ;                       (DH,DL) = ROW,COLUMN OF CHARACTER LP POSN
                        3153    ;                       (CH) = RASTER LINE (0-199)
                        3154    ;                       (BX) = PIXEL COLUMN (0-319,639)
                        3155    ;       (AH)=5  SELECT ACTIVE DISPLAT PAGE (VALID ONLY FOR ALPHA MODES)
                        3156    ;               (AL)=NEW PAGE VALUE (0-7 FOR MODES 0&1, 0-3 FOR MODES 2&3)
```

```
LOC OBJ                 LINE    SOURCE

                        3157    ;       (AH)=6  SCROLL ACTIVE PAGE UP
                        3158    ;               (AL) = NUMBER OF LINES, INPUT LINES BLANKED AT BOTTOM OF WINDOW
                        3159    ;                       AL = 0 MEANS BLANK ENTIRE WINDOW
                        3160    ;               (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL
                        3161    ;               (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL
                        3162    ;               (BH) = ATTRIBUTE TO BE USED ON BLANK LINE
                        3163    ;       (AH)=7  SCROLL ACTIVE PAGE DOWN
                        3164    ;               (AL) = NUMBER OF LINES, INPUT LINES BLANKED AT TOP OF WINDOW
                        3165    ;                       AL = 0 MEANS BLANK ENTIRE WINDOW
                        3166    ;               (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL
                        3167    ;               (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL
                        3168    ;               (BH) = ATTRIBUTE TO BE USED ON BLANK LINE
                        3169    ;
                        3170    ;       CHARACTER HANDLING ROUTINES
                        3171    ;
                        3172    ;       (AH) = 8 READ ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION
                        3173    ;               (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)
                        3174    ;               ON EXIT:
                        3175    ;               (AL) = CHAR READ
                        3176    ;               (AH) = ATTRIBUTE OF CHARACTER READ (ALPHA MODES ONLY)
                        3177    ;       (AH) = 9 WRITE ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION
                        3178    ;               (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)
                        3179    ;               (CX) = COUNT OF CHARACTERS TO WRITE
                        3180    ;               (AL) = CHAR TO WRITE
                        3181    ;               (BL) = ATTRIBUTE OF CHARACTER (ALPHA)/COLOR OF CHAR (GRAPHICS)
                        3182    ;                       SEE NOTE ON WRITE DOT FOR BIT 7 OF BL = 1.
                        3183    ;       (AH) = 10 WRITE CHARACTER ONLY AT CURRENT CURSOR POSITION
                        3184    ;               (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)
                        3185    ;               (CX) = COUNT OF CHARACTERS TO WRITE
                        3186    ;               (AL) = CHAR TO WRITE
                        3187    ;       FOR READ/WRITE CHARACTER INTERFACE WHILE IN GRAPHICS MODE, THE
                        3188    ;               CHARACTERS ARE FORMED FROM A CHARACTER GENERATOR IMAGE
                        3189    ;               MAINTAINED IN THE SYSTEM ROM.  ONLY THE 1ST 128 CHARS
                        3190    ;               ARE CONTAINED THERE.  TO READ/WRITE THE SECOND 128 CHARS,
                        3191    ;               THE USER MUST INITIALIZE THE POINTER AT INTERRUPT 1FH
                        3192    ;               (LOCATION 0007CH) TO POINT TO THE 1K BYTE TABLE CONTAINING
                        3193    ;               THE CODE POINTS FOR THE SECOND 128 CHARS (128-255).
                        3194    ;       FOR WRITE CHAPACTER INTERFACE IN GRAPHICS MODE, THE REPLICATION FACTOR
                        3195    ;               CONTAINED IN (CX) ON ENTRY WILL PRODUCE VALID RESULTS ONLY
                        3196    ;               FOR CHARACTERS CONTAINED ON THE SAME ROW.  CONTINUATION TO
                        3197    ;               SUCCEEDING LINES WILL NOT PRODUCE CORRECTLY.
                        3198    ;
                        3199    ;       GRAPHICS INTERFACE
                        3200    ;       (AH) = 11 SET COLOR PALETTE
                        3201    ;               (BH) = PALLETTE COLOR ID BEING SET (0-127)
                        3202    ;               (BL) = COLOR VALUE TO BE USED WITH THAT COLOR ID
                        3203    ;                  NOTE: FOR THE CURRENT COLOR CARD, THIS ENTRY POINT HAS
                        3204    ;                               MEANING ONLY FOR 320X200 GRAPHICS.
                        3205    ;                       COLOR ID = 0 SELECTS THE BACKGROUND COLOR (0-15)
                        3206    ;                       COLOR ID = 1 SELECTS THE PALLETTE TO BE USED:
                        3207    ;                               0 = GREEN(1)/RED(2)/YELLOW(3)
                        3208    ;                               1 = CYAN(1)/MAGENTA(2)/WHITE(3)
                        3209    ;                       IN 40X25 OR 80X25 ALPHA MODES, THE VALUE SET FOR
                        3210    ;                               PALLETTE COLOR 0 INDICATES THE BORDER COLOR
                        3211    ;                               TO BE USED (VALUES 0-31, WHERE 16-31 SELECT THE
                        3212    ;                               HIGH INTENSITY BACKGROUND SET.
                        3213    ;       (AH) = 12 WRITE DOT
                        3214    ;               (DX) = ROW NUMBER
                        3215    ;               (CX) = COLUMN NUMBER
                        3216    ;               (AL) = COLOR VALUE
                        3217    ;                       IF BIT 7 OF AL = 1, THEN THE COLOR VALUE IS EXCLUSIVE
                        3218    ;                       OR'D WITH THE CURRENT CONTENTS OF THE DOT
                        3219    ;       (AH) = 13 READ DOT
                        3220    ;               (DX) = ROW NUMBER
                        3221    ;               (CX) = COLUMN NUMBER
                        3222    ;               (AL) RETURNS THE DOT READ
                        3223    ;
                        3224    ; ASCII TELETTPE ROUTINE FOR OUTPUT
                        3225    ;
                        3226    ;       (AH) = 14 WRITE TELETYPE
                        3227    ;               (AL) = CHAR TO WRITE
                        3228    ;               (BL) = FOREGROUND COLOR IN GRAPHICS MODE
                        3229    ;               (BH) = DISPLAY PAGE IN ALPHA MODE
                        3230    ;               NOTE -- SCREEN WIDTH IS CONTROLLED BY PREVIOUS MODE SET
                        3231    ;
```



```
LOC OBJ                 LINE    SOURCE

                        3232    ;       (AH) = 15 CURRENT VIDEO STATE
                        3233    ;               RETURNS THE CURRENT VIDEO STATE
                        3234    ;               (AL) = MODE CURRENTLY SET ( SEE AH=0 FOR EXPLANATION)
                        3235    ;               (AH) = NUMBER OF CHARACTER COLUMNS ON SCREEN
                        3236    ;               (BH) = CURRENT ACTIVE DISPLAY PAGE
                        3237    ;
                        3238    ;       CS,SS,DS,ES,BX,CX,DX PRESERVED DURING CALL
                        3239    ;       ALL OTHERS DESTROYED
                        3240    ;--------------------------------------------
                        3241            ASSUME  CS:CODE,DS:DATA,ES:VIDEO_RAM
                        3242
F045                    3243    M1      LABEL   WORD    ; TABLE OF ROUTINES WITHIN VIDEO I/O
F045 FCF0           R   3244            DW      OFFSET  SET_MODE
F047 CFF1           R   3245            DW      OFFSET  SET_CTYPE
F049 F0F1           R   3246            DW      OFFSET  SET_CPOS
F04B 1AF2           R   3247            DW      OFFSET  READ_CURSOR
F04D A9F7           R   3248            DW      OFFSET  READ_LPEN
F04F 30F2           R   3249            DW      OFFSET  ACT_DISP_PAGE
F051 9CF2           R   3250            DW      OFFSET  SCROLL_UP
F053 41F3           R   3251            DW      OFFSET  SCROLL_DOWN
F055 7DF3           R   3252            DW      OFFSET  READ_AC_CURRENT
F057 C3F3           R   3253            DW      OFFSET  WRITE_AC_CURRENT
F059 F6F3           R   3254            DW      OFFSET  WRITE_C_CURRENT
F05B 54F2           R   3255            DW      OFFSET  SET_COLOR
F05D 38F4           R   3256            DW      OFFSET  WRITE_DOT
F05F 27F4           R   3257            DW      OFFSET  READ_DOT
F061 22F7           R   3258            DW      OFFSET  WRITE_TTY
F063 7AF2           R   3259            DW      OFFSET  VIDEO_STATE
  0020                  3260    M1L     EQU     $-M1
                        3261
F065                    3262    VIDEO_IO        PROC    NEAR
                        3263            STI             ; INTERRUPTS BACK ON
F066 FC                 3264            CLD             ; SET DIRECTION FORWARD
F067 06                 3265            PUSH    ES
F068 1E                 3266            PUSH    DS              ; SAVE SEGMENT REGISTERS
F069 52                 3267            PUSH    DX
F06A 51                 3268            PUSH    CX
F06B 53                 3269            PUSH    BX
F06C 56                 3270            PUSH    SI
F06D 57                 3271            PUSH    DI
F06E 50                 3272            PUSH    AX              ; SAVE AX VALUE
F06F 8AC4               3273            MOV     AL,AH           ; GET INTO LOW BYTE
F071 32E4               3274            XOR     AH,AH           ; ZERO TO HIGH BYTE
F073 D1E0               3275            SAL     AX,1            ; *2 FOR TABLE LOOKUP
F075 8BF0               3276            MOV     SI,AX           ; PUT INTO SI FOR BRANCH
F077 3D2000             3277            CMP     AX,M1L          ; TEST FOR WITHIN RANGE
F07A 7204               3278            JB      M2              ; BRANCH AROUND BRANCH
F07C 58                 3279            POP     AX              ; THROW AWAY THE PARAMETER
F07D E94701             3280            JMP     VIDEO_RETURN            ; DO NOTHING IF NOT IN RANGE
F080 B84000         R   3281    M2:     MOV     AX,DATA
F083 8ED8               3282            MOV     DS,AX
F085 B8008B             3283            MOV     AX,0B800H       ; SEGMENT FOR COLOR CARD
F088 883E1000       R   3284            MOV     DI,EQUIP_FLAG   ; GET EQUIPMENT SETTING
F08C 81E73000           3285            AND     DI,30H          ; ISOLATE CRT SWITCHES
F090 83FF30             3286            CMP     DI,30H          ; IS SETTING FOR BW CARD?
F093 7503               3287            JNE     M3
F095 B8008D             3288            MOV     AX,0B000H       ; SEGMENT FOR BW CARD
F098 8EC0               3289    M3:     MOV     ES,AX           ; SET UP TO POINT AT VIDEO RAM AREAS
F09A 58                 3290            POP     AX              ; RECOVER VALUE
F09B 8A264900       R   3291            MOV     AH,CRT_MODE     ; GET CURRENT MODE INTO AH
F09F 2EFFA445F0     R   3292            JMP     WORD PTR CS:[SI+OFFSET M1]
                        3293    VIDEO_IO        ENDP
                        3294    ;--------------------------------------------
                        3295    ; SET_MODE
                        3296    ;       THIS ROUTINE INITIALIZES THE ATTACHMENT TO
                        3297    ;       THE SELECTED MODE.  THE SCREEN IS BLANKED.
                        3298    ; INPUT
                        3299    ;       (AL) = MODE SELECTED (RANGE 0-9)
                        3300    ; OUTPUT
                        3301    ;       NONE
                        3302    ;--------------------------------------------
                        3303
                        3304    ;------ TABLES FOR USE IN SETTING OF MODE
                        3305
F0A4                    3306    VIDEO_PARMS     LABEL   BYTE
                        3307    ;------ INIT_TABLE
```

```
LOC OBJ                  LINE   SOURCE

F0A4 38282D0A1F0619      3308           DB      38H,28H,2DH,0AH,1FH,6,19H       ; SET UP FOR 40X25
F0AB 1C02070607          3309           DB      1CH,2,7,6,7
F0B0 00000000            3310           DB      0,0,0,0
  0010                   3311   M4      EQU     $-VIDEO_PARMS
                         3312
F0B4 71505A0A1F0619      3313           DB      71H,50H,5AH,0AH,1FH,6,19H       ; SET UP FOR 80X25
F0BB 1C02070607          3314           DB      1CH,2,7,6,7
F0C0 00000000            3315           DB      0,0,0,0
                         3316
F0C4 38282D0A7F0664      3317           DB      38H,28H,2DH,0AH,7FH,6,64H       ; SET UP FOR GRAPHICS
F0CB 7002010607          3318           DB      70H,2,1,6,7
F0D0 00000000            3319           DB      0,0,0,0
                         3320
F0D4 61505A0F190619      3321           DB      61H,50H,52H,0FH,19H,6,19H       ; SET UP FOR 80X25 B&W CARD
F0DB 1902000B0C          3322           DB      19H,2,0DH,0BH,0CH
F0E0 00000000            3323           DB      0,0,0,0
                         3324
F0E4                     3325   M5      LABEL   WORD                    ; TABLE OF REGEN LENGTHS
F0E4 0008                3326           DW      2048            ; 40X25
F0E6 0010                3327           DW      4096            ; 80X25
F0E8 0040                3328           DW      16384           ; GRAPHICS
F0EA 0040                3329           DW      16384           ;
                         3330
                         3331   ;------ COLUMNS
F0EC                     3332   M6      LABEL   BYTE
F0EC 2828505028285050    3333           DB      40,40,80,80,40,40,80,80
                         3334
                         3335   ;------ C_REG_TAB
F0F4                     3336   M7      LABEL   BYTE                    ; TABLE OF MODE SETS
F0F4 2C282D292A2E1E29    3337           DB      2CH,28H,2DH,29H,2AH,2EH,1EH,29H ;
                         3338
F0FC                     3339   SET_MODE        PROC    NEAR
F0FC BAD403              3340           MOV     DX,03D4H                ; ADDRESS OF COLOR CARD
F0FF B030                3341           MOV     BL,0                    ; MODE SET FOR COLOR CARD
F101 83FF30              3342           CMP     DI,30H                  ; IS BW CARD INSTALLED
F104 7507                3343           JNE     M8                      ; OK WITH COLOR
F106 B007                3344           MOV     AL,7                    ; INDICATE BW CARD MODE
F108 BAB403              3345           MOV     DX,03B4H                ; ADDRESS OF BW CARD
F10B FEC3                3346           INC     BL                      ; MODE SET FOR BW CARD
F10D 8AE0                3347   M8:     MOV     AH,AL           ; SAVE MODE IN AH
F10F A24900              R 3348           MOV     CRT_MODE,AL     ; SAVE IN GLOBAL VARIABLE
F112 89166300            R 3349           MOV     ADDR_6845,DX    ; SAVE ADDRESS OF BASE
F116 1E                  3350           PUSH    DS              ; SAVE POINTER TO DATA SEGMENT
F117 50                  3351           PUSH    AX              ; SAVE MODE
F118 52                  3352           PUSH    DX              ; SAVE OUTPUT PORT VALUE
F119 83C204              3353           ADD     DX,4            ; POINT TO CONTROL REGISTER
F11C 8AC3                3354           MOV     AL,BL           ; GET MODE SET FOR CARD
F11E EE                  3355           OUT     DX,AL           ; RESET VIDEO
F11F 5A                  3356           POP     DX              ; BACK TO BASE REGISTER
F120 2BC0                3357           SUB     AX,AX           ; SET UP FOR ABS0 SEGMENT
F122 8ED8                3358           MOV     DS,AX           ; ESTABLISH VECTOR TABLE ADDRESSING
                         3359           ASSUME  DS:ABS0
F124 C51E7400            3360           LDS     BX,PARM_PTR     ; GET POINTER TO VIDEO PARMS
F128 58                  3361           POP     AX              ; RECOVER PARMS
                         3362           ASSUME  DS:CODE
F129 B91000              3363           MOV     CX,M4           ; LENGTH OF EACH ROW OF TABLE
F12C 80FC02              3364           CMP     AH,2            ; DETERMINE WHICH ONE TO USE
F12F 7210                3365           JC      M9              ; MODE IS 0 OR 1
F131 0309                3366           ADD     BX,CX           ; MOVE TO NEXT ROW OF IHIT TABLE
F133 80FC04              3367           CMP     AH,4
F136 7209                3368           JC      M9              ; MODE IS 2 OR 3
F138 0309                3369           ADD     BX,CX           ; MOVE TO GRAPHICS ROW OF INIT_TABLE
F13A 80FC07              3370           CMP     AH,7
F13D 7202                3371           JC      M9              ; MODE IS 4,5, OR 6
F13F 0309                3372           ADD     BX,CX           ; MOVE TO BW CARD ROW OF INIT_TABLE
                         3373
                         3374   ;------ BX POINTS TO CORRECT ROW OF INITIALIZATION TABLE
                         3375
F141                     3376   M9:                             ; OUT_INIT
F141 50                  3377           PUSH    AX              ; SAVE MODE IN AH
F142 32E4                3378           XOR     AH,AH           ; AH WILL SERVE AS REGISTER NUMBER DURING LOOP
                         3379
                         3380   ;------ LOOP THROUGH TABLE, OUTPUTTTING REG ADDRESS, THEN VALUE FROM TABLE
                         3381
F144                     3382   M10:                            ; INIT LOOP
F144 8AC4                3383           MOV     AL,AH           ; GET 6845 REGISTER NUMBER

                                                                                                  A-46
```

```
LOC OBJ                  LINE   SOURCE

F146 EE                  3384           OUT     DX,AL
F147 42                  3385           INC     DX              ; POINT TO DATA PORT
F148 FEC4                3386           INC     AH              ; NEXT REGISTER VALUE
F14A 8A07                3387           MOV     AL,[BX]         ; GET TABLE VALUE
F14C EE                  3388           OUT     DX,AL           ; OUT TO CHIP
F14D 43                  3389           INC     BX              ; NEXT IN TABLE
F14E 4A                  3390           DEC     DX              ; BACK TO POINTER REGISTER
F14F E2F3                3391           LOOP    M10             ; DO THE WHOLE TABLE
F151 58                  3392           POP     AX              ; GET MODE BACK
F152 1F                  3393           POP     DS              ; RECOVER SEGMENT VALUE
                         3394           ASSUME  DS:DATA
                         3395
                         3396   ;------ FILL REGEN AREA WITH BLANK
                         3397
F153 33FF                3398           XOR     DI,DI           ; SET UP POINTER FOR REGEN
F155 893E4E00            R 3399         MOV     CRT_START,DI    ; START ADDRESS SAVED IN GLOBAL
F159 C606620000          R 3400         MOV     ACTIVE_PAGE,0   ; SET PAGE VALUE
F15E B90020              3401           MOV     CX,8192         ; NUMBER OF WORDS IN COLOR CARD
F161 80FC04              3402           CMP     AH,4            ; TEST FOR GRAPHICS
F164 720C                3403           JC      M12             ; NO_GRAPHICS_INIT
F166 80FC07              3404           CMP     AH,7            ; TEST FOR BW CARD
F169 7404                3405           JE      M11             ; BW_CARD_INIT
F16B 33C0                3406           XOR     AX,AX           ; FILL FOR GRAPHICS MODE
F16D EB06                3407           JMP     SHORT M13       ; CLEAR_BUFFER
F16F                     3408   M11:                            ; BW_CARD_INIT
F16F B90008              3409           MOV     CX,2048         ; BUFFER SIZE ON BW CARD
F172                     3410   M12:                            ; NO_GRAPHICS_INIT
F172 B80207              3411           MOV     AX,' '+7*256    ; FILL CHAR FOR ALPHA
F175                     3412   M13:                            ; CLEAR_BUFFER
F175 F3                  3413           REP     STOSW           ; FILL THE REGEN BUFFER WITH BLANKS
F176 AB
                         3414
                         3415   ;------ ENABLE VIDEO AND CORRECT PORT SETTING
                         3416
F177 C70666006700        R 3417         MOV     CURSOR_MODE,67H ; SET CURRENT CURSOR MODE
F17D A04900              R 3418         MOV     AL,CRT_MODE     ; GET THE MODE
F180 32E4                3419           XOR     AH,AH           ;  INTO AX REGISTER
F182 8BF0                3420           MOV     SI,AX           ; TABLE POINTER, INDEXED BY MODE
F184 8B166300            R 3421         MOV     DX,ADDR_6845    ; PREPARE TO OUTPUT TO VIDEO ENABLE PORT
F188 83C204              3422           ADD     DX,4
F18B 2E8A84F4F0          R 3423         MOV     AL,CS:[SI+OFFSET M7]
F190 EE                  3424           OUT     DX,AL           ; SET VIDEO ENABLE PORT
F191 A26500              R 3425         MOV     CRT_MODE_SET.AL ; SAVE THAT VALUE
                         3426
                         3427   ;------ DETERMINE NUMBER OF COLUMNS. BOTH FOR ENTIRE DISPLAY
                         3428   ;------ AND THE NUMBER TO BE USED FOR TTY INTERFACE
                         3429
F194 2E8A84ECF0          R 3430         MOV     AL,CS:[SI + OFFSET M6]
F199 32E4                3431           XOR     AH,AH
F19B A34A00              R 3432         MOV     CRT_COLS,AX     ; NUMBER OF COLUMNS IN THIS SCREEN
                         3433
                         3434   ;------ SET CURSOR POSITIONS
                         3435
F19E 81E60E00            3436           AND     SI,0EH          ; WORD OFFSET INTO CLEAR LENGTH TABLE
F1A2 2E8B8CE4F0          R 3437         MOV     CX,CS:[SI + OFFSET M5]  ; LENGTH TO CLEAR
F1A7 890E4C00            R 3438         MOV     CRT_LEN,CX      ; SAVE LENGTH OF CRT -- NOT USED FOR BW
F1AB B90800              3439           MOV     CX,8            ; CLEAR ALL CURSOR POSITIONS
F1AE BF5000              R 3440         MOV     DI,OFFSET CURSOR_POSN
F1B1 1E                  3441           PUSH    DS              ; ESTABLISH SEGMENT
F1B2 07                  3442           POP     ES              ;   ADDRESSING
F1B3 33C0                3443           XOR     AX,AX
F1B5 F3                  3444           REP     STOSW           ; FILL WITH ZEROES
F1B6 AB
                         3445
                         3446   ;------ SET UP OVERSCAN REGISTER
                         3447
F1B7 42                  3448           INC     DX              ; SET OVERSCAN PORT TO A DEFAULT
F1B8 B030                3449           MOV     AL,30H          ; VALUE OF 30H FOR ALL MODES EXCEPT 640X200
F1BA 803E490006          R 3450         CMP     CRT_MODE,6      ; SEE IF THE MODE IS 640X200 BW
F1BF 7502                3451           JNZ     M14             ; IF IT ISNT 640X200, THEN GOTO REGULAR
F1C1 B03F                3452           MOV     AL,3FH          ; IF IT IS 640X200, THEN PUT IN 3FH
F1C3 EE                  3453   M14:    OUT     DX,AL           ; OUTPUT THE CORRECT VALUE TO 3D9 PORT
F1C4 A26600              R 3454         MOV     CRT_PALLETTE,AL ; SAVE THE VALUE FOR FUTURE USE
                         3455
                         3456   ;------ NORMAL RETURN FROM ALL VIDEO RETURNS
                         3457

                                                                                                  A-47
```

```
LOC OBJ                 LINE    SOURCE

F1C7                    3458    VIDEO_RETURN:
F1C7 5F                 3459            POP     DI
F1C8 5E                 3460            POP     SI
F1C9 5B                 3461            POP     BX
F1CA                    3462    M15:                            ; VIDEO_RETURN_C
F1CA 59                 3463            POP     CX
F1CB 5A                 3464            POP     DX
F1CC 1F                 3465            POP     DS
F1CD 07                 3466            POP     ES              ; RECOVER SEGMENTS
F1CE CF                 3467            IRET                    ; ALL DONE
                        3468    SET_MODE        ENDP
                        3469    ;--------------------------------------------
                        3470    ; SET_CTYPE
                        3471    ;       THIS ROUTINE SETS THE CURSOR VALUE
                        3472    ; INPUT
                        3473    ;       (CX) HAS CURSOR VALUE CH-START LINE, CL-STOP LINE
                        3474    ; OUTPUT
                        3475    ;       NONE
                        3476    ;--------------------------------------------
F1CF                    3477    SET_CTYPE       PROC    NEAR
F1CF B40A               3478            MOV     AH,10           ; 6845 REGISTER FOR CURSOR SET
F1D1 890E6000     R     3479            MOV     CURSOR_MODE,CX  ; SAVE IN DATA AREA
F1D5 E80200             3480            CALL    M16             ; OUTPUT CX REG
F1D8 EBED               3481            JMP     VIDEO_RETURN
                        3482
                        3483    ;------ THIS ROUTINE OUTPUTS THE CX REGISTER TO THE 6845 REGS NAMED IN AH
                        3484
F1DA                    3485    M16:
F1DA 8B166300     R     3486            MOV     DX,ADDR_6845    ; ADDRESS REGISTER
F1DE 8AC4               3487            MOV     AL,AH           ; GET VALUE
F1E0 EE                 3488            OUT     DX,AL           ; REGISTER SET
F1E1 42                 3489            INC     DX              ; DATA REGISTER
F1E2 8AC5               3490            MOV     AL,CH           ; DATA
F1E4 EE                 3491            OUT     DX,AL
F1E5 4A                 3492            DEC     DX
F1E6 8AC4               3493            MOV     AL,AH
F1E8 FEC0               3494            INC     AL              ; POINT TO OTHER DATA REGISTER
F1EA EE                 3495            OUT     DX,AL           ; SET FOR SECOND REGISTER
F1EB 42                 3496            INC     DX
F1EC 8AC1               3497            MOV     AL,CL           ; SECOND DATA VALUE
F1EE EE                 3498            OUT     DX,AL
F1EF C3                 3499            RET                     ; ALL DONE
                        3500    SET_CTYPE       ENDP
                        3501    ;--------------------------------------------
                        3502    ; SET_CPOS
                        3503    ;       THIS ROUTINE SETS THE CURRENT CURSOR POSITION TO THE
                        3504    ;       NEW X-Y VALUES PASSED
                        3505    ; INPUT
                        3506    ;       DX - ROW,COLUMN OF NEW CURSOR
                        3507    ;       BH - DISPLAY PAGE OF CURSOR
                        3508    ; OUTPUT
                        3509    ;       CURSOR IS SET AT 6845 IF DISPLAY PAGE IS CURRENT DISPLAY
                        3510    ;--------------------------------------------
F1F0                    3511    SET_CPOS        PROC    NEAR
F1F0 8ACF               3512            MOV     CL,BH
F1F2 32ED               3513            XOR     CH,CH           ; ESTABLISH LOOP COUNT
F1F4 D1E1               3514            SAL     CX,1            ; WORD OFFSET
F1F6 8BF1               3515            MOV     SI,CX           ; USE INDEX REGISTER
F1F8 89945000     R     3516            MOV     [SI+OFFSET CURSOR_POSN],DX      ; SAVE THE POINTER
F1FC 383E6200     R     3517            CMP     ACTIVE_PAGE,BH
F200 7505               3518            JNZ     M17             ; SET_CPOS_RETURN
F202 8BC2               3519            MOV     AX,DX           ; GET ROW/COLUMN TO AX
F204 E80200             3520            CALL    M18             ; CURSOR_SET
F207                    3521    M17:                            ; SET_CPOS_RETURN
F207 EBBE               3522            JMP     VIDEO_RETURN
                        3523    SET_CPOS        ENDP
                        3524
                        3525    ;------ SET CURSOR POSITION, AX HAS ROW/COLUMN FOR CURSOR
                        3526
F209                    3527    M18     PROC    NEAR
F209 E87F00             3528            CALL    POSITION        ; DETERMINE LOCATION IN REGEN BUFFER
F20C 8BC8               3529            MOV     CX,AX
F20E 030E4E00     R     3530            ADD     CX,CRT_START    ; ADD IN THE START ADDRESS FOR THIS PAGE
F212 D1F9               3531            SAR     CX,1            ; DIVIDE BY 2 FOR CHAR ONLY COUNT
F214 B40E               3532            MOV     AH,14           ; REGISTER NUMBER FOR CURSOR
```



```
LOC OBJ                 LINE    SOURCE

F216 E8C1FF             3533            CALL    M16             ; OUTPUT THE VALUE TO THE 6845
F219 C3                 3534            RET
                        3535    M18     ENDP
                        3536    ;--------------------------------------------
                        3537    ; READ_CURSOR
                        3538    ;       THIS ROUTINE READS THE CURRENT CURSOR VALUE FROM THE
                        3539    ;       6845, FORMATS IT, AND SENDS IT BACK TO THE CALLER
                        3540    ; INPUT
                        3541    ;       BH - PAGE OF CURSOR
                        3542    ; OUTPUT
                        3543    ;       DX - ROW, COLUMN OF THE CURRENT CURSOR POSITION
                        3544    ;       CX - CURRENT CURSOR MODE
                        3545    ;--------------------------------------------
F21A                    3546    READ_CURSOR     PROC    NEAR
F21A 8ADF               3547            MOV     BL,BH
F21C 32FF               3548            XOR     BH,BH
F21E D1E3               3549            SAL     BX,1            ; WORD OFFSET
F220 8B975000     R     3550            MOV     DX,[BX+OFFSET CURSOR_POSN]
F224 8B0E6000     R     3551            MOV     CX,CURSOR_MODE
F228 5F                 3552            POP     DI
F229 5E                 3553            POP     SI
F22A 5B                 3554            POP     BX
F22B 58                 3555            POP     AX      ; DISCARD SAVED CX AND DX
F22C 58                 3556            POP     AX
F22D 1F                 3557            POP     DS
F22E 07                 3558            POP     ES
F22F CF                 3559            IRET
                        3560    READ_CURSOR     ENDP
                        3561    ;--------------------------------------------
                        3562    ; ACT_DISP_PAGE
                        3563    ;       THIS ROUTINE SETS THE ACTIVE DISPLAY PAGE, ALLOWING
                        3564    ;       THE FULL USE OF THE RAM SET ASIDE FOR THE VIDEO ATTACHMENT
                        3565    ; INPUT
                        3566    ;       AL HAS THE NEW ACTIVE DISPLAY PAGE
                        3567    ; OUTPUT
                        3568    ;       THE 6845 IS RESET TO DISPLAY THAT PAGE
                        3569    ;--------------------------------------------
F230                    3570    ACT_DISP_PAGE   PROC    NEAR
F230 A26200       R     3571            MOV     ACTIVE_PAGE,AL  ; SAVE ACTIVE PAGE VALUE
F233 8B0E4C00     R     3572            MOV     CX,CRT_LEN      ; GET SAVED LENGTH OF REGEN BUFFER
F237 98                 3573            CBW                     ; CONVERT AL TO WORD
F238 50                 3574            PUSH    AX              ; SAVE PAGE VALUE
F239 F7E1               3575            MUL     CX              ; DISPLAY PAGE TIMES REGEN LENGTH
F23B A34E00       R     3576            MOV     CRT_START,AX    ; SAVE START ADDRESS FOR LATER REQUIREMENTS
F23E 8BC8               3577            MOV     CX,AX           ; START ADDRESS TO CX
F240 D1F9               3578            SAR     CX,1            ; DIVIDE BY 2 FOR 6845 HANDLING
F242 B40C               3579            MOV     AH,12           ; 6845 REGISTER FOR START ADDRESS
F244 E893FF             3580            CALL    M16
F247 5B                 3581            POP     BX              ; RECOVER PAGE VALUE
F248 D1E3               3582            SAL     BX,1            ; *2 FOR WORD OFFSET
F24A 8B875000     R     3583            MOV     AX,[BX + OFFSET CURSOR_POSN]    ; GET CURSOR FOR THIS PAGE
F24E E8B8FF             3584            CALL    M18             ; SET THE CURSOR POSITION
F251 E973FF             3585            JMP     VIDEO_RETURN
                        3586    ACT_DISP_PAGE   ENDP
                        3587    ;--------------------------------------------
                        3588    ; SET COLOR
                        3589    ;       THIS ROUTINE WILL ESTABLISH THE BACKGROUND COLOR, THE OVERSCAN COLOR,
                        3590    ;       AND THE FOREGROUND COLOR SET FOR MEDIUM RESOLUTION GRAPHICS
                        3591    ; INPUT
                        3592    ;       (BH) HAS COLOR ID
                        3593    ;               IF BH=0, THE BACKGROUND COLOR VALUE IS SET
                        3594    ;                       FROM THE LOW BITS OF BL (0-31)
                        3595    ;               IF BH=1, THE PALLETTE SELECTION IS MADE
                        3596    ;                       BASED ON THE LOW BIT OF BL:
                        3597    ;                               0 = GREEN, RED, YELLOW FOR COLORS 1,2,3
                        3598    ;                               1 = BLUE, CYAN, MAGENTA FOR COLORS 1,2,3
                        3599    ;       (BL) HAS THE COLOR VALUE TO BE USED
                        3600    ; OUTPUT
                        3601    ;       THE COLOR SELECTION IS UPDATED
                        3602    ;--------------------------------------------
F254                    3603    SET_COLOR       PROC    NEAR
F254 8B166300     R     3604            MOV     DX,ADDR_6845    ; I/O PORT FOR PALETTE
F258 83C205             3605            ADD     DX,5            ; OVERSCAN PORT
F25B A0660D       R     3606            MOV     AL,CRT_PALLETTE ; GET THE CURRENT PALLETTE VALUE
F25E 0AFF               3607            OR      BH,BH           ; IS THIS COLOR 0?
```

```
LOC OBJ                 LINE    SOURCE

F260 750E               3608            JNZ     M20             ; OUTPUT COLOR 1
                        3609
                        3610    ;------ HANDLE COLOR 0 BY SETTING THE BACKGROUND COLOR
                        3611
F262 24E0               3612            AND     AL,0E0H         ; TURN OFF LOW 5 BITS OF CURRENT
F264 80E31F             3613            AND     BL,01FH         ; TURN OFF HIGH 3 BITS OF INPUT VALUE
F267 0AC3               3614            OR      AL,BL           ; PUT VALUE INTO REGISTER
F269                    3615    M19:                            ; OUTPUT THE PALLETTE
F269 EE                 3616            OUT     DX,AL           ;output color selection to 3d9 port
F26A A26600         R   3617            MOV     CRT_PALLETTE,AL ; SAVE THE COLOR VALUE
F26D E957FF             3618            JMP     VIDEO_RETURN
                        3619
                        3620    ;------ HANDLE COLOR 1 BY SELECTING THE PALLETTE TO BE USED
                        3621
F270                    3622    M20:
F270 240F               3623            AND     AL,00FH         ; TURN OFF PALLETTE SELECT BIT
F272 00EB               3624            SHR     BL,1            ; TEST THE LOW ORDER BIT OF BL
F274 73F3               3625            JNC     M19             ; ALREADY DONE
F276 0C20               3626            OR      AL,20H          ; TURN ON PALLETTE SELECT BIT
F278 EBEF               3627            JMP     M19             ; GO DO IT
                        3628    SET_COLOR       ENDP
                        3629    ;--------------------------------------------
                        3630    ;VIDEO STATE
                        3631    ;  RETURNS THE CURRENT VIDEO STATE IN AX
                        3632    ;  AH = NUMBER OF COLUMNS ON THE SCREEN
                        3633    ;  AL = CURRENT VIDEO MODE
                        3634    ;  BH = CURRENT ACTIVE PAGE
                        3635    ;--------------------------------------------
F27A                    3636    VIDEO_STATE     PROC    NEAR
F27A 8A264A00       R   3637            MOV     AH,BYTE PTR CRT_COLS    ; GET NUMBER OF COLUMNS
F27E A04900         R   3638            MOV     AL,CRT_MODE             ; CURRENT MODE
F281 8A3E6200       R   3639            MOV     BH,ACTIVE_PAGE          ; GET CURRENT ACTIVE PAGE
F285 5F                 3640            POP     DI                      ; RECOVER REGISTERS
F286 5E                 3641            POP     SI                      ;
F287 59                 3642            POP     CX                      ; DISCARD SAVED BX
F288 E93FFF             3643            JMP     M15                     ; RETURN TO CALLER
                        3644    VIDEO_STATE     ENDP
                        3645    ;--------------------------------
                        3646    ; POSITION
                        3647    ;       THIS SERVICE ROUTINE CALCULATES THE REGEN BUFFER ADDRESS
                        3648    ;       OF A CHARACTER IN THE ALPHA MODE
                        3649    ;INPUT
                        3650    ;       AX = ROW, COLUMN POSITION
                        3651    ; OUTPUT
                        3652    ;       AX = OFFSET OF CHAR POSITION IN REGEN BUFFER
                        3653    ;--------------------------------
F288                    3654    POSITION        PROC    NEAR
F288 53                 3655            PUSH    BX              ; SAVE REGISTER
F28C 8BD8               3656            MOV     BX,AX
F28E 8AC4               3657            MOV     AL,AH           ; ROWS TO AL
F290 F6264A00       R   3658            MUL     BYTE PTR CRT_COLS       ; DETERMINE BYTES TO ROW
F294 32FF               3659            XOR     BH,BH
F296 03C3               3660            ADD     AX,BX           ; ADD IN COLUMN VALUE
F298 D1E0               3661            SAL     AX,1            ; * 2 FOR ATTRIBUTE BYTES
F29A 5B                 3662            POP     BX
F29B C3                 3663            RET
                        3664    POSITION        ENDP
                        3665    ;--------------------------------------------
                        3666    ; SCROLL UP
                        3667    ;       THIS ROUTINE MOVES A BLOCK OF CHARACTERS UP
                        3668    ;       ON THE SCREEN
                        3669    ;INPUT
                        3670    ;       (AH) = CURRENT CRT MODE
                        3671    ;       (AL) = NUMBER OF ROWS TO SCROLL
                        3672    ;       (CX) = ROW/COLUMN OF UPPER LEFT CORNER
                        3673    ;       (DX) = ROW/COLUMN OF LOWER RIGHT CORNER
                        3674    ;       (BH) = ATTRIBUTE TO BE USED ON BLANKED LINE
                        3675    ;       (DS) = DATA SECMENT
                        3676    ;       (ES) = REGEN BUFFER SEGMENT
                        3677    ;OUTPUT
                        3678    ;       NONE -- THE REGEN BUFFER IS MODIFIED
                        3679    ;--------------------------------------------
                        3680            ASSUME  CS:CODE,DS:DATA,ES:DATA
F29C                    3681    SCROLL_UP       PROC    NEAR
F29C 8AD8               3682            MOV     BL,AL           ; SAVE LINE COUNT IN BL
F29E 80FC04             3683            CMP     AH,4            ; TEST FOR GRAPHICS MODE
```



```
LOC OBJ                 LINE    SOURCE

F2A1 7208               3684            JC      N1              ; HANDLE SEPARATELY
F2A3 80FC07             3685            CMP     AH,7            ; TEST FOR BW CARD
F2A6 7403               3686            JE      N1
F2A8 E9F301             3687            JMP     GRAPHICS_UP
F2AB                    3688    N1:                             ; UP_CONTINUE
F2AB 53                 3689            PUSH    BX              ; SAVE FILL ATTRIBUTE IN BH
F2AC 8BC1               3690            MOV     AX,CX           ; UPPER LEFT POSITION
F2AE E83900             3691            CALL    SCROLL_POSITION ; DO SETUP FOR SCROLL
F2B1 7433               3692            JZ      N7              ; BLANK_FIELD
F2B3 03F0               3693            ADD     SI,AX           ; FROM ADDRESS
F2B5 8AE6               3694            MOV     AH,DH           ; # ROWS IN BLOCK
F2B7 2AE3               3695            SUB     AH,BL           ; # ROWS TO BE MOVED
F2B9                    3696    N2:                             ; ROW_LOOP
F2B9 E87500             3697            CALL    N10             ; MOVE ONE ROW
F2BC 03F5               3698            ADD     SI,BP
F2BE 03FD               3699            ADD     DI,BP           ; POINT TO NEXT LINE IN BLOCK
F2C0 FECC               3700            DEC     AH              ; COUNT OF LINES TO MOVE
F2C2 75F5               3701            JNZ     N2              ; ROW_LOOP
F2C4                    3702    N3:                             ; CLEAR_ENTRY
F2C4 58                 3703            POP     AX              ; RECOVER ATTRIBUTE IN AH
F2C5 B020               3704            MOV     AL,' '          ; FILL WITH BLANKS
F2C7                    3705    N4:                             ; CLEAR_LOOP
F2C7 E87000             3706            CALL    N11             ; CLEAR THE ROW
F2CA 03FD               3707            ADD     DI,BP           ; POINT TO NEXT LINE
F2CC FECB               3708            DEC     BL              ; COUNTER OF LINES TO SCROLL
F2CE 75F7               3709            JNZ     N4              ; CLEAR_LOOP
F2D0                    3710    N5:                             ; SCROLL_END
F2D0 B84000         R   3711            MOV     AX,DATA         ; GET LOCATION
F2D3 8ED8               3712            MOV     DS,AX
F2D5 803E490007     R   3713            CMP     CRT_MODE,7      ; IS THIS THE BLACK AND WHITE CARD
F2DA 7407               3714            JE      N6              ; IF SO, SKIP THE MODE RESET
F2DC A06500         R   3715            MOV     AL,CRT_MODE_SET ; GET THE VALUE OF THE MODE SET
F2DF BAD803             3716            MOV     DX,03D8H        ; ALWAYS SET COLOR CARD PORT
F2E2 EE                 3717            OUT     DX,AL
F2E3                    3718    N6:                             ; VIDEO_RET_HERE
F2E3 E9E1FE             3719            JMP     VIDEO_RETURN
F2E6                    3720    N7:                             ; BLANK_FIELD
F2E6 8ADE               3721            MOV     BL,DH           ; GET ROW COUNT
F2E8 EB0A               3722            JMP     N3              ; GO CLEAR THAT AREA
                        3723    SCROLL_UP       ENDP
                        3724
                        3725    ;----- HANDLE COMMON SCROLL SET UP HERE
                        3726
F2EA                    3727    SCROLL_POSITION PROC    NEAR
F2EA 803E490002     R   3728            CMP     CRT_MODE,2      ; TEST FOR SPECIAL CASE HERE
F2EF 7219               3729            JB      N9              ; HAVE TO HANDLE 80X25 SEPARATELY
F2F1 803E490003     R   3730            CMP     CRT_MODE,3
F2F6 7712               3731            JA      N9
                        3732
                        3733    ;------ 80X25 COLOR CARD SCROLL
                        3734
F2F8 52                 3735            PUSH    DX
F2F9 BADA03             3736            MOV     DX,3DAH         ; GUARANTEED TO BE COLOR CARD HERE
F2FC 50                 3737            PUSH    AX
F2FD                    3738    N8:                             ; WAIT_DISP_ENABLE
F2FD EC                 3739            IN      AL,DX           ; GET PORT
F2FE A808               3740            TEST    AL,8            ; WAIT FOR VERTICAL RETRACE
F300 74FB               3741            JZ      N8              ; WAIT_DISP_ENABLE
F302 B025               3742            MOV     AL,25H
F304 BAD803             3743            MOV     DX,03D8H
F307 EE                 3744            OUT     DX,AL           ; TURN OFF VIDEO
F308 58                 3745            POP     AX              ; DURING VERTICAL RETRACE
F309 5A                 3746            POP     DX
F30A E87EFF             3747    N9:     CALL    POSITION        ; CONVERT TO REGEN POINTER
F30D 03064E00       R   3748            ADD     AX,CRT_START    ; OFFSET OF ACTIVE PAGE
F311 8BF8               3749            MOV     DI,AX           ; TO ADDRESS FOR SCROLL
F313 8BF0               3750            MOV     SI,AX           ; FROM ADDRESS FOR SCROLL
F315 2BD1               3751            SUB     DX,CX           ; DX = #ROWS, #COLS IN BLOCK
F317 FEC6               3752            INC     DH
F319 FEC2               3753            INC     DL              ; INCREMENT FOR 0 ORIGIN
F31B 32ED               3754            XOR     CH,CH           ; SET HIGH BYTE OF COUNT TO ZERO
F31D 8B2E4A00       R   3755            MOV     BP,CRT_COLS     ; GET NUMBER OF COLUMNS IN DISPLAY
F321 03ED               3756            ADD     BP,BP           ; TIMES 2 FOR ATTRIBUTE BYTE
F323 8AC3               3757            MOV     AL,BL           ; GET LINE COUNT
F325 F6264A00       R   3758            MUL     BYTE PTR CRT_COLS       ; DETERMINE OFFSET TO FROM ADDRESS
F329 03C0               3759            ADD     AX,AX           ; *2 FOR ATTRIBUTE BYTE
```

```
LOC OBJ                 LINE    SOURCE

F328 06                 3760            PUSH    ES              ; ESTABLISH ADDRESSING TO REGEN BUFFER
F32C 1F                 3761            POP     DS              ;  FOR BOTH POINTERS
F320 80FB00             3762            CMP     BL,0            ; 0 SCROLL MEANS BLANK FIELD
F330 C3                 3763            RET                     ; RETURN WITH FLAGS SET
                        3764    SCROLL_POSITION ENDP
                        3765
                        3766    ;------ MOVE_ROW
F331                    3767    N10     PROC    NEAR
F331 8ACA               3768            MOV     CL,DL           ; GET # OF COLS TO MOVE
F333 56                 3769            PUSH    SI
F334 57                 3770            PUSH    DI              ; SAVE START ADDRESS
F335 F3                 3771            REP     MOVSW           ; MOVE THAT LINE ON SCREEN
F336 A5
F337 5F                 3772            POP     DI
F338 5E                 3773            POP     SI              ; RECOVER ADDRESSES
F339 C3                 3774            RET
                        3775    N10     ENDP
                        3776
                        3777    ;------ CLEAR_ROW
F33A                    3778    N11     PROC    NEAR
F33A 8ACA               3779            MOV     CL,DL           ; GET # COLUMNS TO CLEAR
F33C 57                 3780            PUSH    DI
F33D F3                 3781            REP     STOSW           ; STORE THE FILL CHARACTER
F33E AB
F33F 5F                 3782            POP     DI
F340 C3                 3783            RET
                        3784    N11     ENDP
                        3785    ;--------------------------------------
                        3786    ; SCROLL_DOWN
                        3787    ;       THIS ROUTINE MOVES THE CHARACTERS WITHIN A DEFINED
                        3788    ;       BLOCK DOWN ON THE SCREEN, FILLING THE TOP LINES
                        3789    ;       WITH A DEFINED CHARACTER
                        3790    ;INPUT
                        3791    ;       (AH) = CURRENT CRT MODE
                        3792    ;       (AL) = NUMBER OF LINES TO SCROLL
                        3793    ;       (CX) = UPPER LEFT CORNER OF REGION
                        3794    ;       (DX) = LOWER RIGHT CORNER OF REGION
                        3795    ;       (BH) = FILL CHARACTER
                        3796    ;       (DS) = DATA SEGMENT
                        3797    ;       (ES) = REGEN SEGMENT
                        3798    ;OUPUT
                        3799    ;       NONE -- SCREEN IS SCROLLED
                        3800    ;--------------------------------------
F341                    3801    SCROLL_DOWN     PROC    NEAR
F341 FD                 3802            STD                     ; DIRECTION FOR SCROLL DOWN
F342 8ADB               3803            MOV     BL,AL           ; LINE COUNT TO BL
F344 80FC04             3804            CMP     AH,4            ; TEST FOR GRAPHICS
F347 720B               3805            JC      N12
F349 80FC07             3806            CMP     AH,7            ; TEST FOR BW CARD
F34C 7403               3807            JE      N12
F34E E9A601             3808            JMP     GRAPHICS_DOWN
F351                    3809    N12:                            ; CONTINUE_DOWN
F351 53                 3810            PUSH    BX              ; SAVE ATTRIBUTE IN BH
F352 8BC2               3811            MOV     AX,DX           ; LOWER RIGHT CORNER
F354 E893FF             3812            CALL    SCROLL_POSITION ; GET REGEN LOCATION
F357 7420               3813            JZ      N16
F359 2BF0               3814            SUB     SI,AX           ; SI IS FROM ADDRESS
F35B 8AE6               3815            MOV     AH,DH           ; GET TOTAL # ROWS
F350 2AE3               3816            SUB     AH,BL           ; COUNT TO MOVE IN SCROLL
F35F                    3817    N13:
F35F E8CFFF             3818            CALL    N10             ; MOVE ONE ROW
F362 2BF5               3819            SUB     SI,BP
F364 2BF0               3820            SUB     DI,BP
F366 FECC               3821            DEC     AH
F368 75F5               3822            JNZ     N13
F36A                    3823    N14:
F36A 58                 3824            POP     AX              ; RECOVER ATTRIBUTE IN AH
F36B B020               3825            MOV     AL,
F360                    3826    N15:
F36D E8CAFF             3827            CALL    N11             ; CLEAR ONE ROW
F370 2BFD               3828            SUB     DI,BP           ; GO TO NEXT ROW
F372 FECB               3829            DEC     BL
F374 75F7               3830            JNZ     N15
F376 E957FF             3831            JMP     N5                      ; SCROLL_END
F379                    3832    N16:
F379 8ADE               3833            MOV     BL,DH
```



```
LOC OBJ                 LINE    SOURCE

F37B EBED               3834            JMP     N14
                        3835    SCROLL_DOWN     ENDP
                        3836    ;--------------------------------------
                        3837    ; READ_AC_CURRENT
                        3838    ;       THIS ROUTINE READS THE ATTRIBUTE AND CHARACTER AT THE CURRENT
                        3839    ;       CURSOR POSITION AND RETURNS THEM TO THE CALLER
                        3840    ;INPUT
                        3841    ;       (AH) = CURRENT CRT MODE
                        3842    ;       (BH) = DISPLAY PAGE ( ALPHA MODES ONLY )
                        3843    ;       (DS) = DATA SEGMENT
                        3844    ;       (ES) = REGEN SEGMENT
                        3845    ;OUTPUT
                        3846    ;       (AL) = CHAR READ
                        3847    ;       (AH) = ATTRIBUTE READ
                        3848    ;--------------------------------------
                        3849            ASSUME  CS:CODE,DS:DATA,ES:DATA
F37D                    3850    READ_AC_CURRENT PROC    NEAR
F37D 80FC04             3851            CMP     AH,4                    ; IS THIS GRAPHICS
F380 7208               3852            JC      P1
F382 80FC07             3853            CMP     AH,7                    ; IS THIS BW CARD
F385 7403               3854            JE      P1
F387 E9A902             3855            JMP     GRAPHICS_READ
F38A                    3856    P1:                             ; READ_AC_CONTINUE
F38A E81A00             3857            CALL    FIND_POSITION
F380 8BF3               3858            MOV     SI,BX                   ; ESTABLISH ADDRESSING IN SI
                        3859
                        3860    ;------ WAIT FOR HORIZONTAL RETRACE
                        3861
F38F 8B166300       R   3862            MOV     DX,ADDR_6845            ; GET BASE ADDRESS
F393 83C206             3863            ADD     DX,6                    ; POINT AT STATUS PORT
F396 06                 3864            PUSH    ES                      ;
F397 1F                 3865            POP     DS                      ; GET SEGMENT FOR QUICK ACCESS
F398                    3866    P2:                             ; WAIT FOR RETRACE LOW
F398 EC                 3867            IN      AL,DX                   ; GET STATUS
F399 A801               3868            TEST    AL,1                    ; IS HORZ RETRACE LOW
F39B 75FB               3869            JNZ     P2                      ; WAIT UNTIL IT IS
F39D FA                 3870            CLI                             ; NO MORE INTERRUPTS
F39E                    3871    P3:                             ; WAIT FOR RETRACE HIGH
F39E EC                 3872            IN      AL,DX                   ; GET STATUS
F39F A801               3873            TEST    AL,1                    ; IS IT HIGH
F3A1 74FB               3874            JZ      P3                      ; WAIT UNTIL IT IS
F3A3 AD                 3875            LODSW                           ; GET THE CHAR/ATTR
F3A4 E920FE             3876            JMP     VIDEO_RETURN
                        3877    READ_AC_CURRENT ENDP
                        3878
F3A7                    3879    FIND_POSITION   PROC    NEAR
F3A7 8ACF               3880            MOV     CL,BH                   ; DISPLAY PAGE TO CX
F3A9 32E0               3881            XOR     CH,CH
F3AB 8BF1               3882            MOV     SI,CX                   ; MOVE TO SI FOR INDEX
F3AD 01E6               3883            SAL     SI,1                    ; * 2 FOR WORD OFFSET
F3AF 8B845000       R   3884            MOV     AX,[SI+ OFFSET CURSOR_POSN]     ; GET ROW/COLUMN OF THAT PAGE
F3B3 3308               3885            XOR     BX,BX                   ; SET START ADDRESS TO ZERO
F3B5 E306               3886            JCXZ    P5                      ; NO_PAGE
F3B7                    3887    P4:                             ; PAGE_LOOP
F3B7 031E4C00       R   3888            ADD     BX,CRT_LEN              ; LENGTH OF BUFFER
F3BB E2FA               3889            LOOP    P4
F3BD                    3890    P5:                             ; NO_PAGE
F3BD E8CBFE             3891            CALL    POSITION                ; DETERMINE LOCATION IN REGEN
F3C0 0308               3892            ADD     BX,AX                   ; ADD TO START OF REGEN
F3C2 C3                 3893            RET
                        3894    FIND_POSITION   ENDP
                        3895    ;--------------------------------------
                        3896    ;WRITE_AC_CURRENT
                        3897    ;       THIS ROUTINE WRITES THE ATTRIBUTE AND CHARACTER AT
                        3898    ;       THE CURRENT CURSOR POSITION
                        3899    ;INPUT
                        3900    ;       (AH) = CURRENT CRT MODE
                        3901    ;       (BH) = DISPLAY PAGE
                        3902    ;       (CX) = COUNT OF CHARACTERS TO WRITE
                        3903    ;       (AL) = CHAR TO WRITE
                        3904    ;       (BL) = ATTRIBUTE OF CHAR TO WRITE
                        3905    ;       (DS) = DATA SEGMENT
                        3906    ;       (ES) = REGEN SEGMENT
                        3907    ;OUTPUT
                        3908    ;       NONE
                        3909    ;--------------------------------------
```

```
LOC OBJ                 LINE    SOURCE


F3C3                    3910    WRITE_AC_CURRENT        PROC    NEAR
F3C3 80FC04             3911            CMP     AH,4                    ; IS THIS GRAPHICS
F3C6 7208               3912            JC      P6
F3C8 80FC07             3913            CMP     AH,7                    ; IS THIS BW CARD
F3CB 7403               3914            JE      P6
F3CD E98101             3915            JMP     GRAPHICS_WRITE
F3D0                    3916    P6:                                     ; WRITE_AC_CONTINUE
F3D0 8AE3               3917            MOV     AH,BL                   ; GET ATTRIBUTE TO AH
F3D2 50                 3918            PUSH    AX                      ; SAVE ON STACK
F3D3 51                 3919            PUSH    CX                      ; SAVE WRITE COUNT
F3D4 E800FF             3920            CALL    FIND_POSITION
F3D7 8BFB               3921            MOV     DI,BX                   ; ADDRESS TO DI REGISTER
F3D9 59                 3922            POP     CX                      ; WRITE COUNT
F3DA 58                 3923            POP     BX                      ; CHARACTER IN BX REG
F3DB                    3924    P7:                                     ; WRITE_LOOP
                        3925
                        3926    ;------ WAIT FOR HORIZONTAL RETRACE
                        3927
F3DB 8B166300       R   3928            MOV     DX,ADDR_6845            ; GET BASE ADDRESS
F3DF 83C206             3929            ADD     DX,6                    ; POINT AT STATUS PORT
F3E2                    3930    P8:
F3E2 EC                 3931            IN      AL,DX                   ; GET STATUS
F3E3 A801               3932            TEST    AL,1                    ; IS IT LOW
F3E5 75FB               3933            JNZ     P8                      ; WAIT UNTIL IT IS
F3E7 FA                 3934            CLI                             ; NO MORE INTERRUPTS
F3E8                    3935    P9:
F3E8 EC                 3936            IN      AL,DX                   ; GET STATUS
F3E9 A801               3937            TEST    AL,1                    ; IS IT HIGH
F3EB 74FB               3938            JZ      P9                      ; WAIT UNTIL IT IS
F3ED 8BC3               3939            MOV     AX,BX                   ; RECOVER THE CHAR/ATTR
F3EF AB                 3940            STOSW                           ; PUT THE CHAR/ATTR
F3F0 FB                 3941            STI                             ; INTERRUPTS BACK ON
F3F1 E2E8               3942            LOOP    P7                      ;  AS MANY TIMES AS REQUESTED
F3F3 E9D1FD             3943            JMP     VIDEO_RETURN
                        3944    WRITE_AC_CURRENT        ENDP
                        3945    ;----------------------------------------
                        3946    ;WRITE_C_CURRENT
                        3947    ;       THIS ROUTINE WRITES THE CHARACTER AT
                        3948    ;       THE CURRENT CURSOR POSITION, ATTRIBUTE UNCHANGED
                        3949    ;INPUT
                        3950    ;       (AH) = CURRENT CRT MODE
                        3951    ;       (BH) = DISPLAY PAGE
                        3952    ;       (CX) = COUNT OF CHARACTERS TO WRITE
                        3953    ;       (AL) = CHAR TO WRITE
                        3954    ;       (DS) = DATA SEGMENT
                        3955    ;       (ES) = REGEN SEGMENT
                        3956    ;OUTPUT
                        3957    ;       NONE
                        3958    ;----------------------------------------
F3F6                    3959    WRITE_C_CURRENT PROC    NEAR
F3F6 80FC04             3960            CMP     AH,4                    ; IS THIS GRAPHICS
F3F9 7208               3961            JC      P10
F3FB 80FC07             3962            CMP     AH,7                    ; IS THIS BW CARD
F3FE 7403               3963            JE      P10
F400 E97E01             3964            JMP     GRAPHICS_WRITE
F403                    3965    P10:
F403 50                 3966            PUSH    AX                      ; SAVE ON STACK
F404 51                 3967            PUSH    CX                      ; SAVE WRITE COUNT
F405 E89FFF             3968            CALL    FIND_POSITION
F408 8BFB               3969            MOV     DI,BX                   ; ADDRESS TO DI
F40A 59                 3970            POP     CX                      ; WRITE COUNT
F40B 5B                 3971            POP     BX                      ; BL HAS CHAR TO WRITE
F40C                    3972    P11:                                    ; WRITE_LOOP
                        3973
                        3974    ;------ WAIT FOR HORIZONTAL RETRACE
                        3975
F40C 8B166300       R   3976            MOV     DX,ADDR_6845            ; GET BASE ADDRESS
F410 83C206             3977            ADD     DX,6                    ; POINT AT STATUS PORT
F413                    3978    P12:
F413 EC                 3979            IN      AL,DX                   ; GET STATUS
F414 A801               3980            TEST    AL,1                    ; IS IT LOW
F416 75FB               3981            JNZ     P12                     ; WAIT UNTIL IT IS
F418 FA                 3982            CLI                             ; NO MORE INTERRUPTS
F419                    3983    P13:
F419 EC                 3984            IN      AL,DX                   ; GET STATUS
F41A A801               3985            TEST    AL,1                    ; IS IT HIGH
```



```
LOC OBJ                 LINE    SOURCE


F41C 74FB               3986            JZ      P13                     ; WAIT UNTIL IT IS
F41E 8AC3               3987            MOV     AL,BL                   ; RECOVER CHAR
F420 AA                 3988            STOSB                           ; PUT THE CHAR/ATTR
F421 47                 3989            INC     DI                      ; BUMP POINTER PAST ATTRIBUTE
F422 E2E8               3990            LOOP    P11                     ;  AS MANY TIMES AS REQUESTED
F424 E9A0FD             3991            JMP     VIDEO_RETURN
                        3992    WRITE_C_CURRENT ENDP
                        3993    ;----------------------------------------
                        3994    ; READ DOT  -- WRITE DOT
                        3995    ; THESE ROUTINES WILL WRITE A DOT, OR READ THE
                        3996    ;  DOT AT THE INDICATED LOCATION
                        3997    ; ENTRY --
                        3998    ;   DX = ROW (0-199)    (THE ACTUAL VALUE DEPENDS ON THE MODE)
                        3999    ;   CX = COLUMN ( 0-639) ( THE VALUES ARE NOT RANGE CHECKED )
                        4000    ;   AL = DOT VALUE TO WRITE (1,2 OR 4 BITS DEPENDING ON MODE,
                        4001    ;       REQ'D FOR WRITE DOT ONLY, RIGHT JUSTIFIED)
                        4002    ;       BIT 7 OF AL = 1 INDICATES XOR THE VALUE INTO THE LOCATION
                        4003    ;   DS = DATA SEGMENT
                        4004    ;   ES = REGEN SEGMENT
                        4005    ;
                        4006    ; EXIT
                        4007    ;       AL = DOT VALUE READ, RIGHT JUSTIFIED, READ ONLY
                        4008    ;----------------------------------------------
                        4009            ASSUME  CS:CODE,DS:DATA,ES:DATA
F427                    4010    READ_DOT        PROC    NEAR
F427 E83100             4011            CALL    R3              ; DETERMINE BYTE POSITION OF DOT
F42A 268A04             4012            MOV     AL,ES:[SI]              ; GET THE BYTE
F42D 22C4               4013            AND     AL,AH           ; MASK OFF THE OTHER BITS IN THE BYTE
F42F D2E0               4014            SHL     AL,CL           ; LEFT JUSTIFT THE VALUE
F431 8ACE               4015            MOV     CL,DH           ; GET NUMBER OF BITS IN RESULT
F433 D2C0               4016            ROL     AL,CL           ; RIGHT JUSTIFT THE RESULT
F435 E98FFD             4017            JMP     VIDEO_RETURN    ; RETURN FROM VIDEO IO
F438                    4018    READ_DOT        ENDP
                        4019
F438                    4020    WRITE_DOT       PROC    NEAR
F438 50                 4021            PUSH    AX              ; SAVE DOT VALUE
F439 50                 4022            PUSH    AX              ;  TWICE
F43A E81E00             4023            CALL    R3              ; DETERMINE BYTE POSITION OF THE DOT
F43D D2E8               4024            SHR     AL,CL           ; SHIFT TO SET UP THE BITS FOR OUTPUT
F43F 22C4               4025            AND     AL,AH           ; STRIP OFF THE OTHER BITS
F441 268A0C             4026            MOV     CL,ES:[SI]      ; GET THE CURRENT BYTE
F444 5B                 4027            POP     BX              ; RECOVER XOR FLAG
F445 F6C380             4028            TEST    BL,80H          ; IS IT ON
F448 750D               4029            JNZ     R2              ; YES, XOR THE DOT
F44A F6D4               4030            NOT     AH              ; SET THE MASK TO REMOVE THE INDICATED BITS
F44C 22CC               4031            AND     CL,AH
F44E 0AC1               4032            OR      AL,CL           ; OR IN THE NEW VALUE OF THOSE BITS
F450                    4033    R1:                             ; FINISH_DOT
F450 268804             4034            MOV     ES:[SI],AL      ; RESTORE THE BYTE IN MEMORY
F453 58                 4035            POP     AX
F454 E970FD             4036            JMP     VIDEO_RETURN    ; RETURN FROM VIDEO IO
F457                    4037    R2:                             ; XOR_DOT
F457 32C1               4038            XOR     AL,CL           ; EXCLUSIVE OR THE DOTS
F459 EBF5               4039            JMP     R1              ; FINISH UP THE WRITING
                        4040    WRITE_DOT       ENDP
                        4041    ;----------------------------------------------
                        4042    ; THIS SUBROUTINE DETERMINES THE REGEN BYTE LOCATION OF THE
                        4043    ; INDICATED ROW COLUMN VALUE IN GRAPHICS MODE.
                        4044    ; ENTRY --
                        4045    ;  DX = ROW VALUE (0-199)
                        4046    ;  CX = COLUMN VALUE (0-639)
                        4047    ; EXIT --
                        4048    ;  SI = OFFSET INTO REGEN BUFFER FOR BYTE OF INTEREST
                        4049    ;  AH = MASK TO STRIP OFF THE BITS OF INTEREST
                        4050    ;  CL = BITS TO SHIFT TO RIGHT JUSTIFY THE MASK IN AH
                        4051    ;  DH = # BITS IN RESULT
                        4052    ;----------------------------------------------
F45B                    4053    R3      PROC    NEAR
F45B 53                 4054            PUSH    BX              ; SAVE BX DURING OPERATION
F45C 50                 4055            PUSH    AX              ; WILL SAVE AL DURING OPERATION
                        4056
                        4057    ;------ DETERMINE 1ST BYTE IN IDICATED ROW BY MULTIPLYING ROW VALUE BY 40
                        4058    ;------ ( LOW BIT OF ROW DETERMINES EVEN/ODD, 80 BYTES/ROW
                        4059
F45D B028               4060            MOV     AL,40
F45F 52                 4061            PUSH    DX              ; SAVE ROW VALUE
```

```
LOC OBJ                  LINE    SOURCE

F460 80E2FE              4062            AND     DL,0FEH         ; STRIP OFF ODD/EVEN BIT
F463 F6E2                4063            MUL     DL              ; AX HAS ADDRESS OF 1ST BYTE OF INDICATED ROW
F465 5A                  4064            POP     DX              ; RECOVER IT
F466 F6C201              4065            TEST    DL,1            ; TEST FOR EVEN/ODD
F469 7403                4066            JZ      R4              ; JUMP IF EVEN ROW
F46B 050020              4067            ADD     AX,2000H        ; OFFSET TO LOCATION OF ODD ROWS
F46E                     4068    R4:                             ; EVEN_ROW
F46E 8BF0                4069            MOV     SI,AX           ; MOVE POINTER TO SI
F470 58                  4070            POP     AX              ; RECOVER AL VALUE
F471 8B01                4071            MOV     DX,CX           ; COLUMN VALUE TO DX
                         4072
                         4073    ;------ DETERMINE GRAPHICS MODE CURRENTLY IN EFFECT
                         4074
                         4075    ; SET UP THE REGISTERS ACCORDING TO THE MODE
                         4076    ;  CH = MASK FOR LOW OF COLUMN ADDRESS ( 7/3 FOR HIGH/MED RES)
                         4077    ;  CL = # OF ADDRESS BITS IN COLUMN VALUE ( 3/2 FOR H/M)
                         4078    ;  BL = MASK TO SELECT BITS FROM POINTED BYTE (80H/C0H FOR H/M)
                         4079    ;  BH = NUMBER OF VALID BITS IN POINTED BYTE ( 1/2 FOR H/M)
                         4080
F473 BBC002              4081            MOV     BX,2C0H
F476 B90203              4082            MOV     CX,302H         ; SET PARMS FOR MED RES
F479 803E490006      R   4083            CMP     CRT_MODE,6
F47E 7206                4084            JC      R5              ; HANDLE IF MED ARES
F480 BB8001              4085            MOV     BX,180H
F483 B90307              4086            MOV     CX,703H         ; SET PARMS FOR HIGH RES
                         4087
                         4088    ;------ DETERMINE BIT OFFSET IN BYTE FROM COLUMN MASK
F486                     4089    R5:
F486 22EA                4090            AND     CH,DL           ; ADDRESS OF PEL WITHIN BYTE TO CH
                         4091
                         4092    ;------ DETERMINE BYTE OFFSET FOR THIS LOCATION IN COLUMN
                         4093
F488 03EA                4094            SHR     DX,CL           ; SHIFT BY CORRECT AMOUNT
F48A 03F2                4095            ADD     SI,DX           ; INCREMENT THE POINTER
F48C 8AF7                4096            MOV     DH,BH           ; GET THE # OF BITS IN RESULT TO DH
                         4097
                         4098    ;------ MULTIPLY BH (VALID BITS IN BYTE) BY CH (BIT OFFSET)
                         4099
F48E 2AC9                4100            SUB     CL,CL           ; ZERO INTO STORAGE LOCATION
F490                     4101    R6:
F490 D0C8                4102            ROR     AL,1            ; LEFT JUSTIFY THE VALUE IN AL (FOR WRITE)
F492 02CD                4103            ADD     CL,CH           ; ADD IN THE BIT OFFSET VALUE
F494 FECF                4104            DEC     BH              ; LOOP CONTROL
F496 75F8                4105            JNZ     R6              ; ON EXIT, CL HAS SHIFT COUNT TO RESTORE BITS
F498 8AE3                4106            MOV     AH,BL           ; GET MASK TO AH
F49A D2EC                4107            SHR     AH,CL           ; MOVE THE MASK TO CORRECT LOCATION
F49C 5B                  4108            POP     BX              ; RECOVER REG
F49D C3                  4109            RET                     ; RETURN WITH EVERYTHING SET UP
                         4110    R3      ENDP
                         4111    ;-----------------------------------------
                         4112    ; SCROLL UP
                         4113    ;  THIS ROUTINE SCROLLS UP THE INFORMATION ON THE CRT
                         4114    ; ENTRY --
                         4115    ;  CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL
                         4116    ;  DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL
                         4117    ;   BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS
                         4118    ;  BH = FILL VALUE FOR BLANKED LINES
                         4119    ;  AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE FIELD)
                         4120    ;  DS = DATA SEGMENT
                         4121    ;  ES = REGEN SEGMENT
                         4122    ; EXIT --
                         4123    ;  NOTHING, THE SCREEN IS SCROLLED
                         4124    ;-----------------------------------------
F49E                     4125    GRAPHICS_UP     PROC    NEAR
F49E 8AD8                4126            MOV     BL,AL   ; SAVE LINE COUNT IN BL
F4A0 8BC1                4127            MOV     AX,CX   ; GET UPPER LEFT POSITION INTO AX REG
                         4128
                         4129    ;------ USE CHARACTER SUBROUTINE FOR POSITIONING
                         4130    ;------ ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE
                         4131
F4A2 E86A02              4132            CALL    GRAPH_POSN
F4A5 8BF8                4133            MOV     DI,AX           ; SAVE RESULT AS DESTINATION ADDRESS
                         4134
                         4135    ;------ DETERMINE SIZE OF WINDOW
                         4136
```



```
LOC OBJ                  LINE    SOURCE

F4A7 2B01                4137            SUB     DX,CX
F4A9 81C20101            4138            ADD     DX,101H         ; ADJUST VALUES
F4AD D0E6                4139            SAL     DH,1            ; MULTIPLY # ROWS BY 4 SINCE 8 VERT DOTS/CHAR
F4AF D0E6                4140            SAL     DH,1            ;  AND EVEN/ODD ROWS
                         4141
                         4142    ;------ DETERMINE CRT MODE
                         4143
F4B1 803E490006      R   4144            CMP     CRT_MODE,6      ; TEST FOR MEDIUM RES
F4B6 7304                4145            JNC     R7              ; FIND_SOURCE
                         4146
                         4147    ;------ MEDIUM RES UP
F4B8 D0E2                4148            SAL     DL,1            ; # COLUMNS * 2, SINCE 2 BYTES/CHAR
F4BA D1E7                4149            SAL     DI,1            ; OFFSET *2 SINCE 2 BYTES/CHAR
                         4150
                         4151    ;------ DETERMINE THE SOURCE ADDRESS IN THE BUFFER
F4BC                     4152    R7:                             ; FIND_SOURCE
F4BC 06                  4153            PUSH    ES              ; GET SEGMENTS BOTH POINTING TO REGEN
F4BD 1F                  4154            POP     DS
F4BE 2AED                4155            SUB     CH,CH           ; ZERO TO HIGH OF COUNT REG
F4C0 D0E3                4156            SAL     BL,1            ; MULTIPLY NUMBER OF LINES BY 4
F4C2 D0E3                4157            SAL     BL,1
F4C4 7420                4158            JZ      R11             ; IF ZERO, THEN BLANK ENTIRE FIELD
F4C6 8AC3                4159            MOV     AL,BL           ; GET NUMBER OF LINES IN AL
F4C8 B450                4160            MOV     AH,80           ; 80 BYTES/ROW
F4CA F6E4                4161            MUL     AH              ; DETERMINE OFFSET TO SOURCE
F4CC 8BF7                4162            MOV     SI,DI           ; SET UP SOURCE
F4CE 03F0                4163            ADD     SI,AX           ;  ADD IN OFFSET TO IT
F4D0 8AE6                4164            MOV     AH,DH           ; NUMBER OF ROWS IN FIELD
F4D2 2AE3                4165            SUB     AH,BL           ; DETERMINE NUMBER TO MOVE
                         4166
                         4167    ;------ LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS
F4D4                     4168    R8:                             ; ROW_LOOP
F4D4 E88000              4169            CALL    R17             ; MOVE ONE ROW
F4D7 81EEB01F            4170            SUB     SI,2000H-80     ; MOVE TO NEXT ROW
F4DB 81EFB01F            4171            SUB     DI,2000H-80
F4DF FECC                4172            DEC     AH              ; NUMBER OF ROWS TO MOVE
F4E1 75F1                4173            JNZ     R8              ; CONTINUE TILL ALL MOVED
                         4174
                         4175    ;------ FILL IN THE VACATED LINE(S)
F4E3                     4176    R9:                             ; CLEAR_ENTRY
F4E3 8AC7                4177            MOV     AL,BH           ; ATTRIBUTE TO FILL WITH
F4E5                     4178    R10:
F4E5 E88800              4179            CALL    R18             ; CLEAR THAT ROW
F4E8 81EFB01F            4180            SUB     DI,2000H-80     ; POINT TO NEXT LINE
F4EC FECB                4181            DEC     BL              ; NUMBER OF LINES TO FILL
F4EE 75F5                4182            JNZ     R10             ; CLEAR_LOOP
F4F0 E904FC              4183            JMP     VIDEO_RETURN    ; EVERYTHING DONE
                         4184
F4F3                     4185    R11:                            ; BLANK_FIELD
F4F3 8ADE                4186            MOV     BL,DH           ; SET BLANK COUNT TO EVERYTHING IN FIELD
F4F5 E8EC                4187            JMP     R9              ; CLEAR THE FIELD
                         4188    GRAPHICS_UP     ENDP
                         4189    ;-----------------------------------------
                         4190    ; SCROLL DOWN
                         4191    ;  THIS ROUTINE SCROLLS DOWN THE INFORMATION ON THE CRT
                         4192    ; ENTRY --
                         4193    ;  CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL
                         4194    ;  DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL
                         4195    ;   BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS
                         4196    ;  BH = FILL VALUE FOR BLANKED LINES
                         4197    ;  AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE FIELD)
                         4198    ;  DS = DATA SEGMENT
                         4199    ;  ES = REGEN SEGMENT
                         4200    ; EXIT --
                         4201    ;  NOTHING, THE SCREEN IS SCROLLED
                         4202    ;-----------------------------------------
                         4203
F4F7                     4204    GRAPHICS_DOWN   PROC    NEAR
F4F7 FD                  4205            STD             ; SET DIRECTION
F4F8 8AD8                4206            MOV     BL,AL   ; SAVE LINE COUNT IN BL
F4FA 8BC2                4207            MOV     AX,DX   ; GET LOWER RIGHT POSITION INTO AX REG
                         4208
                         4209    ;------ USE CHARACTER SUBROUTINE FOR POSITIONING
                         4210    ;------ ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE
                         4211
F4FC E81100              4212            CALL    GRAPH_POSN
```

```
LOC OBJ                 LINE    SOURCE

F4FF 8BF8               4213            MOV     DI,AX           ; SAVE RESULT AS DESTINATION ADDRESS
                        4214
                        4215    ;------ DETERMINE SIZE OF WINDOW
                        4216
F501 2B01               4217            SUB     DX,CX
F503 81C20101           4218            ADD     DX,101H         ; ADJUST VALUES
F507 00E6               4219            SAL     DH,1            ; MULTIPLY # ROWS BY 4 SINCE 8 VERT DOTS/CHAR
F509 00E6               4220            SAL     DH,1            ;  AND EVEN/ODD ROWS
                        4221
                        4222    ;------ DETERMINE CRT MODE
                        4223
F50B 803E490006     R   4224            CMP     CRT_MODE,6      ; TEST FOR MEDIUM RES
F510 7305               4225            JNC     R12             ; FIND_SOURCE_DOWN
                        4226
                        4227    ;------ MEDIUM RES DOWN
F512 00E2               4228            SAL     DL,1            ; # COLUMNS * 2, SINCE 2 BYTES/CHAR (OFFSET OK)
F514 01E7               4229            SAL     DI,1            ; OFFSET *2 SINCE 2 BYTES/CHAR
F516 47                 4230            INC     DI              ; POINT TO LAST BYTE
                        4231
                        4232    ;------ DETERMINE THE SOURCE ADDRESS IN THE BUFFER
F517                    4233    R12:                            ; FIND_SOURCE_DOWN
F517 06                 4234            PUSH    ES              ; BOTH SEGMENTS TO REGEN
F518 1F                 4235            POP     DS
F519 2AED               4236            SUB     CH,CH           ; ZERO TO HIGH OF COUNT REG
F51B 81C7F000           4237            ADD     DI,240          ; POINT TO LAST ROW OF PIXELS
F51F 00E3               4238            SAL     BL,1            ; MULTIPLY NUMBER OF LINES BY 4
F521 00E3               4239            SAL     BL,1
F523 742E               4240            JZ      R16             ; IF ZERO, THEN BLANK ENTIRE FIELD
F525 8AC3               4241            MOV     AL,BL           ; GET NUMBER OF LINES IN AL
F527 B450               4242            MOV     AH,80           ; 80 BYTES/ROW
F529 F6E4               4243            MUL     AH              ; DETERMINE OFFSET TO SOURCE
F52B 8BF7               4244            MOV     SI,DI           ; SET UP SOURCE
F52D 2BF0               4245            SUB     SI,AX           ;  SUBTRACT THE OFFSET
F52F 8AE6               4246            MOV     AH,DH           ; NUMBER OF ROWS IN FIELD
F531 2AE3               4247            SUB     AH,BL           ; DETERMINE NUMBER TO MOVE
                        4248
                        4249    ;------ LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS
F533                    4250    R13:                            ; ROW_LOOP_DOWN
F533 E82100             4251            CALL    R17             ; MOVE ONE ROW
F536 81EE5020           4252            SUB     SI,2000H+80     ; MOVE TO NEXT ROW
F53A 81EF5020           4253            SUB     DI,2000H+80
F53E FECC               4254            DEC     AH              ; NUMBER OF ROWS TO MOVE
F540 75F1               4255            JNZ     R13             ; CONTINUE TILL ALL MOVED
                        4256
                        4257    ;------ FILL IN THE VACATED LINE(S)
F542                    4258    R14:                            ; CLEAR_ENTRY_DOWN
F542 8AC7               4259            MOV     AL,BH           ; ATTRIBUTE TO FILL WITH
F544                    4260    R15:                            ; CLEAR_LOOP_DOWN
F544 E82900             4261            CALL    R18             ; CLEAR A ROW
F547 81EF5020           4262            SUB     DI,2000H+80     ; POINT TO NEXT LINE
F54B FECB               4263            DEC     BL              ; NUMBER OF LINES TO FILL
F54D 75F5               4264            JNZ     R15             ; CLEAR_LOOP_DOWN
F54F FC                 4265            CLD                     ; RESET THE DIRECTION FLAG
F550 E974FC             4266            JMP     VIDEO_RETURN    ; EVERYTHING DONE
                        4267
F553                    4268    R16:                            ; BLANK_FIELD_DOWN
F553 8ADE               4269            MOV     BL,DH           ; SET BLANK COUNT TO EVERYTHING IN FIELD
F555 EBEB               4270            JMP     R14             ; CLEAR THE FIELD
                        4271    GRAPHICS_DOWN   ENDP
                        4272
                        4273    ;------ ROUTINE TO MOVE ONE ROW OF INFORMATION
                        4274
F557                    4275    R17     PROC    NEAR
F557 8ACA               4276            MOV     CL,DL           ; NUMBER OF BYTES IN THE ROW
F559 56                 4277            PUSH    SI
F55A 57                 4278            PUSH    DI              ; SAVE POINTERS
F55B F3                 4279            REP     MOVSB           ; MOVE THE EVEN FIELD
F55C A4
F55D 5F                 4280            POP     DI
F55E 5E                 4281            POP     SI
F55F 81C60020           4282            ADD     SI,2000H
F563 81C70020           4283            ADD     DI,2000H                ; POINT TO THE ODD FIELD
F567 56                 4284            PUSH    SI
F568 57                 4285            PUSH    DI              ; SAVE THE POINTERS
F569 8ACA               4286            MOV     CL,DL           ; COUNT BACK
```



```
LOC OBJ                 LINE    SOURCE

F56B F3                 4287            REP     MOVSB           ; MOVE THE ODD FIELD
F56C A4
F56D 5F                 4288            POP     DI
F56E 5E                 4289            POP     SI              ; POINTERS BACK
F56F C3                 4290            RET                     ; RETURN TO CALLER
                        4291    R17     ENDP
                        4292
                        4293    ;------ CLEAR A SINGLE ROW
                        4294
F570                    4295    R18     PROC    NEAR
F570 8ACA               4296            MOV     CL,DL           ; NUMBER OF BYTES IN FIELD
F572 57                 4297            PUSH    DI              ; SAVE POINTER
F573 F3                 4298            REP     STOSB           ; STORE THE NEW VALUE
F574 AA
F575 5F                 4299            POP     DI              ; POINTER BACK
F576 81C70020           4300            ADD     DI,2000H                ; POINT TO ODD FIELD
F57A 57                 4301            PUSH    DI
F57B 8ACA               4302            MOV     CL,DL
F57D F3                 4303            REP     STOSB           ; FILL THE ODD FILELD
F57E AA
F57F 5F                 4304            POP     DI
F580 C3                 4305            RET                     ; RETURN TO CALLER
                        4306    R18     ENDP
                        4307    ;--------------------------------------------
                        4308    ; GRAPHICS WRITE
                        4309    ;  THIS ROUTINE WRITES THE ASCII CHARACTER TO THE CURRENT
                        4310    ;  POSITION ON THE SCREEN.
                        4311    ; ENTRY --
                        4312    ;  AL = CHARACTER TO WRITE
                        4313    ;  BL = COLOR ATTRIBUTE TO BE USED FOR FOREGROUND COLOR
                        4314    ;       IF BIT 7 IS SET, THE CHAR IS XOR'D INTO THE REGEN BUFFER
                        4315    ;       (0 IS USED FOR THE BACKGROUND COLOR)
                        4316    ;  CX = NUMBER OF CHARS TO WRITE
                        4317    ;  DS = DATA SEGMENT
                        4318    ;  ES = REGEN SEGMENT
                        4319    ; EXIT --
                        4320    ;  NOTHING IS RETURNED
                        4321    ;
                        4322    ; GRAPHICS READ
                        4323    ;   THIS ROUTINE READS THE ASCII CHARACTER AT THE CURRENT CURSOR
                        4324    ;   POSITION ON THE SCREEN BY MATCHING THE DOTS ON THE SCREEN TO THE
                        4325    ;   CHARACTER GENERATOR CODE POINTS
                        4326    ; ENTRY --
                        4327    ;  NONE  ( 0 IS ASSUMED AS THE BACKGROUND COLOR
                        4328    ; EXIT --
                        4329    ;  AL = CHARACTER READ AT THAT POSITION (0 RETURNED IF NONE FOUND)
                        4330    ;
                        4331    ; FOR BOTH ROUTINES, THE IMAGES USED TO FORM CHARS ARE CONTAINED IN ROM
                        4332    ;  FOR THE 1ST 128 CHARS.  TO ACCESS CHARS IN THE SECOND HALF, THE USER
                        4333    ;  MUST INITIALIZE THE VECTOR AT INTERRUPT 1FH (LOCATION 0007CH) TO
                        4334    ;  POINT TO THE USER SUPPLIED TABLE OF GRAPHIC IMAGES (8X8 BOXES).
                        4335    ;  FAILURE TO DO SO WILL CAUSE IN STRANGE RESULTS
                        4336    ;--------------------------------------------
                        4337            ASSUME  CS:CODE,DS:DATA,ES:DATA
F581                    4338    GRAPHICS_WRITE  PROC    NEAR
F581 B400               4339            MOV     AH,0            ; ZERO TO HIGH OF CODE POINT
F583 50                 4340            PUSH    AX              ; SAVE CODE POINT VALUE
                        4341
                        4342    ;------ DETERMINE POSITION IN REGEN BUFFER TO PUT CODE POINTS
                        4343
F584 E86501             4344            CALL    S26             ; FIND LOCATION IN REGEN BUFFER
F587 8BF8               4345            MOV     DI,AX           ; REGEN POINTER IN DI
                        4346
                        4347    ;------ DETERMINE REGION TO GET CODE POINTS FROM
                        4348
F589 58                 4349            POP     AX              ; RECOVER CODE POINT
F58A 3C80               4350            CMP     AL,80H          ; IS IT IN SECOND HALF
F58C 7306               4351            JAE     S1              ; YES
                        4352
                        4353    ;------ IMAGE IS IN FIRST HALF, CONTAINED IN ROM
                        4354
F58E BE6EFA             4355            MOV     SI,0FA6EH       ; OFFSET CRT_CHAR_GEN-OFFSET OF IMAGES
F591 0E                 4356            PUSH    CS                      ; SAVE SEGMENT ON STACK
F592 EB0F               4357            JMP     SHORT S2                ; DETERMINE_MODE
                        4358
```

```
LOC OBJ                 LINE    SOURCE

                        4359    ;------ IMAGE IS IN SECOND HALF, IN USER RAM
                        4360
F594                    4361    S1:                             ; EXTEND_CHAR
F594 2C80               4362            SUB     AL,80H          ; ZERO ORIGIN FOR SECOND HALF
F596 1E                 4363            PUSH    DS              ; SAVE DATA POINTER
F597 28F6               4364            SUB     SI,SI
F599 8EDE               4365            MOV     DS,SI           ; ESTABLISH VECTOR ADDRESSING
                        4366            ASSUME  DS:ABS0
F59B C5367C00           4367            LDS     SI,EXT_PTR      ; GET THE OFFSET OF THE TABLE
F59F 8CDA               4368            MOV     DX,DS           ; GET THE SEGMENT OF THE TABLE
                        4369            ASSUME  DS:DATA
F5A1 1F                 4370            POP     DS              ; RECOVER DATA SEGMENT
F5A2 52                 4371            PUSH    DX              ; SAVE TABLE SEGMENT ON STACK
                        4372
                        4373    ;------ DETERMINE GRAPHICS MODE IN OPERATION
                        4374
F5A3                    4375    S2:                             ; DETERMINE_MODE
F5A3 D1E0               4376            SAL     AX,1            ; MULTIPLY CODE POINT
F5A5 D1E0               4377            SAL     AX,1            ;  VALUE BY 8
F5A7 D1E0               4378            SAL     AX,1
F5A9 03F0               4379            ADD     SI,AX           ; SI HAS OFFSET OF DESIRED CODES
F5AB 803E490006   R     4380            CMP     CRT_MODE,6
F5B0 1F                 4381            POP     DS              ; RECOVER TABLE POINTER SEGMENT
F5B1 722C               4382            JC      S7              ; TEST FOR MEDIUM RESOLUTION MODE
                        4383
                        4384    ;------ HIGH RESOLUTION MODE
F5B3                    4385    S3:                             ; HIGH_CHAR
F5B3 57                 4386            PUSH    DI              ; SAVE REGEN POINTER
F5B4 56                 4387            PUSH    SI              ; SAVE CODE POINTER
F5B5 B604               4388            MOV     DH,4            ; NUMBER OF TIMES THROUGH LOOP
F5B7                    4389    S4:
F5B7 AC                 4390            LODSB                   ; GET BYTE FROM CODE POINTS
F5B8 F6C380             4391            TEST    BL,80H          ; SHOULD WE USE THE FUNCTION
F5BB 7516               4392            JNZ     S6              ;  TO PUT CHAR IN
F5BD AA                 4393            STOSB                   ; STORE IN REGEN BUFFER
F5BE AC                 4394            LODSB
F5BF                    4395    S5:                             ;
F5BF 26885FF1F          4396            MOV     ES:[DI+2000H-1],AL      ; STORE IN SECOND HALF
F5C4 83C74F             4397            ADD     DI,79           ; MOVE TO NEXT ROW IN REGEN
F5C7 FECE               4398            DEC     DH              ; DONE WITH LOOP
F5C9 75EC               4399            JNZ     S4
F5CB 5E                 4400            POP     SI
F5CC 5F                 4401            POP     DI              ; RECOVER REGEN POINTER
F5CD 47                 4402            INC     DI              ; POINT TO NEXT CHAR POSITION
F5CE E2E3               4403            LOOP    S3              ; MORE CHARS TO WRITE
F5D0 E9F4FB             4404            JMP     VIDEO_RETURN
                        4405
F5D3                    4406    S6:
F5D3 26320F             4407            XOR     AL,ES:[DI]      ; EXCLUSIVE OR WITH CURRENT
F5D6 AA                 4408            STOSB                   ; STORE THE CODE POINT
F5D7 AC                 4409            LODSB                   ; AGAIN FOR ODD FIELD
F5D8 26385FF1F          4410            XOR     AL,ES:[DI+2000H-1]      ;
F5DD EBE0               4411            JMP     S5              ; BACK TO MAINSTREAM
                        4412
                        4413    ;------ MEDIUM RESOLUTION WRITE
F5DF                    4414    S7:                             ; MED_RES_WRITE
F5DF 8AD3               4415            MOV     DL,BL           ; SAVE HIGH COLOR BIT
F5E1 D1E7               4416            SAL     DI,1            ; OFFSET*2 SINCE 2 BYTES/CHAR
F5E3 E8D100             4417            CALL    S19             ; EXPAND BL TO FULL WORD OF COLOR
F5E6                    4418    S8:                             ; MED_CHAR
F5E6 57                 4419            PUSH    DI              ; SAVE REGEN POINTER
F5E7 56                 4420            PUSH    SI                      ; SAVE THE CODE POINTER
F5E8 B604               4421            MOV     DH,4            ; NUMBER OF LOOPS
F5EA                    4422    S9:
F5EA AC                 4423            LODSB                   ; GET CODE POINT
F5EB E80E00             4424            CALL    S21             ; DOUBLE UP ALL THE BITS
F5EE 23C3               4425            AND     AX,BX           ; CONVERT THEM TO FOREGROUND COLOR ( 0 BACK )
F5F0 F6C280             4426            TEST    DL,80H          ; IS THIS XOR FUNCTION
F5F3 7407               4427            JZ      S10             ; NO, STORE IT IN AS IT IS
F5F5 263225             4428            XOR     AH,ES:[DI]      ; DO FUNCTION WITH HALF
F5F8 26324501           4429            XOR     AL,ES:[DI+1]    ;  AND WITH OTHER HALF
F5FC                    4430    S10:                            ;
F5FC 268825             4431            MOV     ES:[DI],AH      ; STORE FIRST BYTE
F5FF 26884501           4432            MOV     ES:[DI+1],AL    ; STORE SECOND BYTE
F603 AC                 4433            LODSB                   ; GET CODE POINT
F604 E8C500             4434            CALL    S21
```



```
LOC OBJ                 LINE    SOURCE

F607 23C3               4435            AND     AX,BX           ; CONVERT TO COLOR
F609 F6C280             4436            TEST    DL,80H          ; AGAIN, IS THIS XOR FUNCTION
F60C 740A               4437            JZ      S11             ; NO, JUST STORE THE VALUES
F60E 2632A50020         4438            XOR     AH,ES:[DI+2000H]        ; FUNCTION WITH FIRST HALF
F613 2632850120         4439            XOR     AL,ES:[DI+2001H]        ; AND WITH SECOND HALF
F618                    4440    S11:                            ;
F618 2688A50020         4441            MOV     ES:[DI+2000H],AH
F61D 2688850120         4442            MOV     ES:[DI+2000H+1],AL      ; STORE IN SECOND PORTION OF BUFFER
F622 83C750             4443            ADD     DI,80           ; POINT TO NEXT LOCATION
F625 FECE               4444            DEC     DH
F627 75C1               4445            JNZ     S9              ; KEEP GOING
F629 5E                 4446            POP     SI              ; RECOVER CODE POINTER
F62A 5F                 4447            POP     DI              ; RECOVER REGEN POINTER
F62B 83C702             4448            ADD     DI,2            ; POINT TO NEXT CHAR POSITION
F62E E2B6               4449            LOOP    S8              ; MORE TO WRITE
F630 E994FB             4450            JMP     VIDEO_RETURN
                        4451    GRAPHICS_WRITE  ENDP
                        4452    ;--------------------------------------
                        4453    ; GRAPHICS READ
                        4454    ;--------------------------------------
F633                    4455    GRAPHICS_READ   PROC    NEAR
F633 E80600             4456            CALL    S26             ; CONVERTED TO OFFSET IN REGEN
F636 8BF0               4457            MOV     SI,AX           ; SAVE IN SI
F638 83EC08             4458            SUB     SP,8            ; ALLOCATE SPACE TO SAVE THE READ CODE POINT
F63B 8BEC               4459            MOV     BP,SP           ; POINTER TO SAVE AREA
                        4460
                        4461    ;------ DETERMINE GRAPHICS MODES
                        4462
F63D 803E490006   R     4463            CMP     CRT_MODE,6
F642 06                 4464            PUSH    ES
F643 1F                 4465            POP     DS              ; POINT TO REGEN SEGMENT
F644 721A               4466            JC      S13             ; MEDIUM RESOLUTION
                        4467
                        4468    ;------ HIGH RESOLUTION READ
                        4469
                        4470    ;------ GET VALUES FROM REGEN BUFFER AND CONVERT TO CODE POINT
F646 B604               4471            MOV     DH,4            ; NUMBER OF PASSES
F648                    4472    S12:
F648 8A04               4473            MOV     AL,[SI]         ; GET FIRST BYTE
F64A 884600             4474            MOV     [BP],AL         ; SAVE IN STORAGE AREA
F64D 45                 4475            INC     BP              ; NEXT LOCATION
F64E 8A840020           4476            MOV     AL,[SI+2000H]   ; GET LOWER REGION BYTE
F652 884600             4477            MOV     [BP],AL         ; ADJUST AND STORE
F655 45                 4478            INC     BP
F656 83C650             4479            ADD     SI,80           ; POINTER INTO REGEN
F659 FECE               4480            DEC     DH              ; LOOP CONTROL
F65B 75EB               4481            JNZ     S12             ; DO IT SOME MORE
F65D EB1790             4482            JMP     S15             ; GO MATCH THE SAVED CODE POINTS
                        4483
F660                    4484    ;------ MEDIUM RESOLUTION READ
F660 D1E6               4485    S13:                            ; MED_RES_READ
F662 B604               4486            SAL     SI,1            ; OFFSET*2 SINCE 2 BYTES/CHAR
F664                    4487            MOV     DH,4            ; NUMBER OF PASSES
F664 E88800             4488    S14:
F667 81C60020           4489            CALL    S23             ; GET PAIR BYTES FROM REGEN INTO SINGLE SAVE
F66B E88100             4490            ADD     SI,2000H        ; GO TO LOWER REGION
F66E 81EE801F           4491            CALL    S23             ; GET THIS PAIR INTO SAVE
F672 FECE               4492            SUB     SI,2000H-80     ; ADJUST POINTER BACK INTO UPPER
F674 75EE               4493            DEC     DH
                        4494            JNZ     S14             ; KEEP GOING UNTIL ALL 8 DONE
                        4495
F676                    4496    ;-------- SAVE AREA HAS CHARACTER IN IT, MATCH IT
F676 BF6EFA             4497    S15:                            ; FIND_CHAR
F679 0E                 4498            MOV     DI,0FA6EH       ; OFFSET CRT_CHAR_GEN-ESTABLISH ADDRESSING
F67A 07                 4499            PUSH    CS
F67B 83E008             4500            POP     ES                      ; CODE POINTS IN CS
F67E 88F5               4501            SUB     BP,8            ; ADJUST POINTER TO BEGINNING OF SAVE AREA
F680 FC                 4502            MOV     SI,BP
F681 B000               4503            CLD                     ; ENSURE DIRECTION
F683                    4504            MOV     AL,0            ; CURRENT CODE POINT BEING MATCHED
F683 16                 4505    S16:
F684 1F                 4506            PUSH    SS              ; ESTABLISH ADDRESSING TO STACK
F685 BA8000             4507            POP     DS              ; FOR THE STRING COMPARE
F688                    4508            MOV     CX,128          ; NUMBER TO TEST AGAINST
F688 56                 4509    S17:
                        4510            PUSH    SI              ; SAVE SAVE AREA POINTER
```

```
LOC OBJ                 LINE    SOURCE

F689 57                 4511            PUSH    DI              ; SAVE CODE POINTER
F68A B90800             4512            MOV     CX,8            ; NUMBER OF BYTES TO MATCH
F68D F3                 4513            REPE    CMPSB           ; COMPARE THE 8 BYTES
F68E A6
F68F 5F                 4514            POP     DI              ; RECOVER THE POINTERS
F690 5E                 4515            POP     SI
F691 741E               4516            JZ      S18             ; IF ZERO FLAG SET, THEN MATCH OCCURRED
F693 FEC0               4517            INC     AL              ; NO MATCH, MOVE ON TO NEXT
F695 83C708             4518            ADD     DI,8            ; NEXT CODE POINT
F698 4A                 4519            DEC     DX              ; LOOP CONTROL
F699 75ED               4520            JNZ     S17             ; DO ALL OF THEM
                        4521
                        4522    ;------ CHAR NOT MATCHED, MIGHT BE IN USER SUPPLIED SECOND HALF
                        4523
F69B 3C00               4524            CMP     AL,0            ; AL<> 0 IF ONLY 1ST HALF SCANNED
F69D 7412               4525            JE      S18             ; IF = 0, THEN ALL HAS BEEN SCANNED
F69F 2BC0               4526            SUB     AX,AX
F6A1 8ED8               4527            MOV     DS,AX           ; ESTABLISH ADDRESSING TO VECTOR
                        4528            ASSUME  DS:ABS0
F6A3 C43E7C00           4529            LES     DI,EXT_PTR      ; GET POINTER
F6A7 8CC0               4530            MOV     AX,ES           ; SEE IF THE POINTER REALLY EXISTS
F6A9 08C7               4531            OR      AX,DI           ; IF ALL 0, THEN DOESN'T EXIST
F6AB 7404               4532            JZ      S18             ; NO SENSE LOOKING
F6AD B080               4533            MOV     AL,128          ; ORIGIN FOR SECOND HALF
F6AF EB02               4534            JMP     S16             ; GO BACK AND TRY FOR IT
                        4535            ASSUME  DS:DATA
                        4536
                        4537    ;------ CHARACTER IS FOUND ( AL=0 IF NOT FOUND )
F6B1                    4538    S18:
F6B1 83C408             4539            ADD     SP,8            ; READJUST THE STACK, THROW AWAY SAVE
F6B4 E910F8             4540            JMP     VIDEO_RETURN    ; ALL DONE
                        4541    GRAPHICS_READ   ENDP
                        4542    ;--------------------------------------------
                        4543    ; EXPAND_MED_COLOR
                        4544    ;  THIS ROUTINE EXPANDS THE LOW 2 BITS IN BL TO
                        4545    ;  FILL THE ENTIRE BX REGISTER
                        4546    ; ENTRY --
                        4547    ;  BL = COLOR TO BE USED ( LOW 2 BITS )
                        4548    ; EXIT --
                        4549    ;  BX = COLOR TO BE USED ( 8 REPLICATIONS OF THE 2 COLOR BITS )
                        4550    ;--------------------------------------------
F6B7                    4551    S19     PROC    NEAR
F6B7 80E303             4552            AND     BL,3            ; ISOLATE THE COLOR BITS
F6BA 8AC3               4553            MOV     AL,BL           ; COPY TO AL
F6BC 51                 4554            PUSH    CX              ; SAVE REGISTER
F6BD B90300             4555            MOV     CX,3            ; NUMBER OF TIMES TO DO THIS
F6C0                    4556    S20:
F6C0 D0E0               4557            SAL     AL,1
F6C2 D0E0               4558            SAL     AL,1            ; LEFT SHIFT BY 2
F6C4 0AD8               4559            OR      BL,AL           ; ANOTHER COLOR VERSION INTO BL
F6C6 E2F8               4560            LOOP    S20             ; FILL ALL OF BL
F6C8 8AFB               4561            MOV     BH,BL           ; FILL UPPER PORTION
F6CA 59                 4562            POP     CX              ; REGISTER BACK
F6CB C3                 4563            RET                     ; ALL DONE
                        4564    S19     ENDP
                        4565    ;--------------------------------------------
                        4566    ; EXPAND_BYTE
                        4567    ;  THIS ROUTINE TAKES THE BYTE IN AL AND DOUBLES ALL
                        4568    ;  OF THE BITS, TURNING THE 8 BITS INTO 16 BITS.
                        4569    ;  THE RESULT IS LEFT IN AX
                        4570    ;--------------------------------------------
F6CC                    4571    S21     PROC    NEAR
F6CC 52                 4572            PUSH    DX              ; SAVE REGISTERS
F6CD 51                 4573            PUSH    CX
F6CE 53                 4574            PUSH    BX
F6CF BA0000             4575            MOV     DX,0            ; RESULT REGISTER
F6D2 B90100             4576            MOV     CX,1            ; MASK REGISTER
F6D5                    4577    S22:
F6D5 8BC8               4578            MOV     BX,AX           ; BASE INTO TEMP
F6D7 23D9               4579            AND     BX,CX           ; USE MASK TO EXTRACT A BIT
F6D9 0BD3               4580            OR      DX,BX           ; PUT INTO RESULT REGISTER
F6DB D1E0               4581            SHL     AX,1
F6DD D1E1               4582            SHL     CX,1            ; SHIFT BASE AND MASK BY 1
F6DF 8BC8               4583            MOV     BX,AX           ; BASE TO TEMP
F6E1 23D9               4584            AND     BX,CX           ; EXTRACT THE SAME BIT
F6E3 0BD3               4585            OR      DX,BX           ; PUT INTO RESULT
```



```
LOC OBJ                 LINE    SOURCE

F6E5 D1E1               4586            SHL     CX,1            ; SHIFT ONLY MASK NOW, MOVING TO NEXT BASE
F6E7 73EC               4587            JNC     S22             ; USE MASK BIT COMING OUT TO TERMINATE
F6E9 8BC2               4588            MOV     AX,DX           ; RESULT TO PARM REGISTER
F6EB 5B                 4589            POP     BX
F6EC 59                 4590            POP     CX              ; RECOVER REGISTERS
F6ED 5A                 4591            POP     DX
F6EE C3                 4592            RET                     ; ALL DONE
                        4593    S21     ENDP
                        4594    ;--------------------------------------------
                        4595    ; MED_READ_BYTE
                        4596    ; THIS ROUTINE WILL TAKE 2 BYTES FROM THE REGEN BUFFER,
                        4597    ;  COMPARE AGAINST THE CURRENT FOREGROUND COLOR, AND PLACE
                        4598    ;  THE CORRESPONDING ON/OFF BIT PATTERN INTO THE CURRENT
                        4599    ;  POSITION IN THE SAVE AREA
                        4600    ; ENTRY --
                        4601    ;  SI,DS = POINTER TO REGEN AREA OF INTEREST
                        4602    ;  BX = EXPANDED FOREGROUND COLOR
                        4603    ;  BP = POINTER TO SAVE AREA
                        4604    ; EXIT --
                        4605    ;  BP IS INCREMENT AFTER SAVE
                        4606    ;--------------------------------------------
F6EF                    4607    S23     PROC    NEAR
F6EF 8A24               4608            MOV     AH,[SI]         ; GET FIRST BYTE
F6F1 8A4401             4609            MOV     AL,[SI+1]       ; GET SECOND BYTE
F6F4 B900C0             4610            MOV     CX,0C000H       ; 2 BIT MASK TO TEST THE ENTRIES
F6F7 B200               4611            MOV     DL,0            ; RESULT REGISTER
F6F9                    4612    S24:
F6F9 85CI               4613            TEST    AX,CX           ; IS THIS SECTION BACKGROUND?
F6FB F8                 4614            CLC                     ; CLEAR CARRY IN HOPES THAT IT IS
F6FC 7401               4615            JZ      S25             ; IF ZERO, IT IS BACKGROUND
F6FE F9                 4616            STC                     ; WASN'T, SO SET CARRY
F6FF D0D2               4617    S25:    RCL     DL,1            ; MOVE THAT BIT INTO THE RESULT
F701 D1E9               4618            SHR     CX,1
F703 D1E9               4619            SHR     CX,1            ; MOVE THE MASK TO THE RIGHT BY 2 BITS
F705 73F2               4620            JNC     S24             ; DO IT AGAIN IF MASK DIDN'T FALL OUT
F707 885600             4621            MOV     [BP],DL         ; STORE RESULT IN SAVE AREA
F70A 45                 4622            INC     BP              ; ADJUST POINTER
F70B C3                 4623            RET                     ; ALL DONE
                        4624    S23     ENDP
                        4625    ;--------------------------------------------
                        4626    ; VG_POSITION
                        4627    ;  THIS ROUTINE TAKES THE CURSOR POSITION CONTAINED IN
                        4628    ;  THE MEMORY LOCATION, AND CONVERTS IT INTO AN OFFSET
                        4629    ;  INTO THE REGEN BUFFER, ASSUMING ONE BYTE/CHAR.
                        4630    ;  FOR MEDIUM RESOLUTION GRAPHICS, THE NUMBER MUST
                        4631    ;  BE DOUBLED.
                        4632    ; ENTRY -- NO REGISTERS,MEMORY LOCATION CURSOR_POSN IS USED
                        4633    ; EXIT--
                        4634    ;  AX CONTAINS OFFSET INTO REGEN BUFFER
                        4635    ;--------------------------------------------
F70C                    4636    S26     PROC    NEAR
F70C A15000         R   4637            MOV     AX,CURSOR_POSN          ; GET CURRENT CURSOR
F70F                    4638    GRAPH_POSN      LABEL   NEAR
F70F 53                 4639            PUSH    BX                      ; SAVE REGISTER
F710 8BD8               4640            MOV     BX,AX                   ; SAVE A COPY OF CURRENT CURSOR
F712 8AC4               4641            MOV     AL,AH                   ; GET ROWS TO AL
F714 F6264A00       R   4642            MUL     BYTE PTR CRT_COLS       ; MULTIPLY BY BYTES/COLUMN
F718 D1E0               4643            SHL     AX,1                    ; MULTIPLY * 4 SINCE 4 ROWS/BYTE
F71A D1E0               4644            SHL     AX,1
F71C 2AFF               4645            SUB     BH,BH                   ; ISOLATE COLUMN VALUE
F71E 03C3               4646            ADD     AX,BX                   ; DETERMINE OFFSET
F720 5B                 4647            POP     BX                      ; RECOVER POINTER
F721 C3                 4648            RET                             ; ALL DONE
                        4649    S26     ENDP
                        4650    ;--------------------------------------------
                        4651    ; WRITE_TTY
                        4652    ; THIS INTERFACE PROVIDES A TELETYPE LIKE INTERFACE TO THE
                        4653    ;   VIDEO CARD.  THE INPUT CHARACTER IS WRITTEN TO THE CURRENT
                        4654    ;   CURSOR POSITION, AND THE CURSOR IS MOVED TO THE NEXT POSITION.
                        4655    ;   IF THE CURSOR LEAVES THE LAST COLUMN OF THE FIELD, THE COLUMN
                        4656    ;   IS SET TO ZERO, AND THE ROW VALUE IS INCREMENTED.  IF THE ROW
                        4657    ;   ROW VALUE LEAVES THE FIELD, THE CURSOR IS PLACED ON THE LAST ROW,
                        4658    ;   FIRST COLUMN, AND THE ENTIRE SCREEN IS SCROLLED UP ONE LINE.
                        4659    ;   WHEN THE SCREEN IS SCROLLED UP, THE ATTRIBUTE FOR FILLING THE
                        4660    ;   NEWLY BLANKED LINE IS READ FROM THE CURSOR POSITION ON THE PREVIOUS
                        4661    ;   LINE BEFORE THE SCROLL.  IN CHARACTER MODE.  IN GRAPHICS MODE,
```

```
LOC OBJ                 LINE    SOURCE

                        4662    ;   THE 0 COLOR IS USED.
                        4663    ; ENTRY --
                        4664    ;   (AH) = CURRENT CRT MODE
                        4665    ;   (AL) = CHARACTER TO BE WRITTEN
                        4666    ;           NOTE THAT BACK SPACE, CAR RET, BELL AND LINE FEED ARE HANDLED
                        4667    ;           AS COMMANDS RATHER THAN AS DISPLAYABLE GRAPHICS
                        4668    ;   (BL) = FOREGROUND COLOR FOR CHAR WRITE IF CURRENTLY IN A GRAPHICS MODE
                        4669    ; EXIT --
                        4670    ;     ALL REGISTERS SAVED
                        4671    ;----------------------------------------
                        4672            ASSUME  CS:CODE,DS:DATA
F722                    4673    WRITE_TTY       PROC    NEAR
F722 50                 4674            PUSH    AX              ; SAVE REGISTERS
F723 50                 4675            PUSH    AX              ; SAVE CHAR TO WRIYE
F724 B403               4676            MOV     AH,3
F726 CD10               4677            INT     10H             ; READ THE CURRENT CURSOR POSITION
F728 58                 4678            POP     AX              ; RECOVER CHAR
                        4679
                        4680    ;------ DX NOW HAS THE CURRENT CURSOR POSITION
                        4681                                    *
F729 3C08               4682            CMP     AL,8            ; IS IT A BACKSPACE
F728 7459               4683            JE      U8              ; BACK_SPACE
F720 3C0D               4684            CMP     AL,0DH          ; IS IT CARRIAGE RETURN
F72F 745E               4685            JE      U9              ; CAR_RET
F731 3C0A               4686            CMP     AL,0AH          ; IS IT A LINE FEED
F733 745E               4687            JE      U10             ; LINE_FEED
F735 3C07               4688            CMP     AL,07H          ; IS IT A BELL
F737 7461               4689            JE      U11             ; BELL
                        4690
                        4691    ;------ WRITE THE CHAR TO THE SCREEN
                        4692
F739 8A3E6200       R   4693            MOV     BH,ACTIVE_PAGE  ; GET THE CURRENT ACTIVE PAGE
F730 B40A               4694            MOV     AH,10           ; WRITE CHAR ONLY
F73F B90100             4695            MOV     CX,1            ; ONLY ONE CHAR
F742 CD10               4696            INT     10H             ; WRITE THE CHAR
                        4697
                        4698    ;------ POSITION THE CURSOR FOR NEXT CHAR
                        4699
F744 FEC2               4700            INC     DL
F746 3A164A00       R   4701            CMP     DL,BYTE PTR CRT_COLS    ; TEST FOR COLUMN OVERFLOW
F74A 7536               4702            JNZ     U7              ; SET_CURSOR
F74C B200               4703            MOV     DL,0            ; COLUMN FOR CURSOR
F74E 80FE18             4704            CMP     DH,24
F751 7520               4705            JNZ     U6              ; SET_CURSOR_INC
                        4706
                        4707    ;------ SCROLL REQUIRED
F753                    4708    U1:
                        4709
F753 B402               4710            MOV     AH,2
F755 B700               4711            MOV     BH,0
F757 CD10               4712            INT     10H             ; SET THE CURSOR
                        4713
                        4714    ;------ DETERMINE VALUE TO FILL WITH DURING SCROLL
                        4715
F759 A04900         R   4716            MOV     AL,CRT_MODE     ; GET THE CURRENT MODE
F75C 3C04               4717            CMP     AL,4
F75E 7206               4718            JC      U2              ; READ-CURSOR
F760 3C07               4719            CMP     AL,7
F762 B700               4720            MOV     BH,0            ; FILL WITH BACKGROUND
F764 7506               4721            JNE     U3              ; SCROLL-UP
                        4722
F766                    4723    U2:                             ; READ-CURSOR
F766 B408               4724            MOV     AH,8
F768 CD10               4725            INT     10H             ; READ CHAR/ATTR AT CURRENT CURSOR
F76A 8AFC               4726            MOV     BH,AH           ; STORE IN BH
                        4727
F76C                    4728    U3:                             ; SCROLL-UP
F76C B80106             4729            MOV     AX,0601H        ; SCROLL ONE LINE
F76F B90000             4730            MOV     CX,0            ; UPPER LEFT CORNER
F772 8618               4731            MOV     DH,24           ; LOWER RIGHT ROW
F774 8A164A00       R   4732            MOV     DL,BYTE PTR CRT_COLS    ; LOWER RIGHT COLUMN
F778 FECA               4733            DEC     DL
F77A                    4734    U4:                             ; VIDEO-CALL-RETURN
F77A CD10               4735            INT     10H             ; SCROLL UP THE SCREEN
F77C                    4736    U5:                             ; TTY-RETURN
F77C 58                 4737            POP     AX              ; RESTORE THE CHARACTER
```



```
LOC OBJ                 LINE    SOURCE

F77D E947FA             4738            JMP     VIDEO_RETURN    ; RETURN TO CALLER
                        4739
F780                    4740    U6:                             ; SET-CURSOR-INC
F780 FEC6               4741            INC     DH              ; NEXT ROW
F782                    4742    U7:                             ; SET-CURSOR
F782 B402               4743            MOV     AH,2
F784 EBF4               4744            JMP     U4              ; ESTABLISH THE NEW CURSOR
                        4745
                        4746    ;------ BACK SPACE FOUND
                        4747
F786                    4748    U8:
F786 80FA00             4749            CMP     DL,0            ; ALREADY AT END OF LINE
F789 74F7               4750            JE      U7              ; SET_CURSOR
F78B FECA               4751            DEC     DL              ; NO -- JUST MOVE IT BACK
F78D EBF3               4752            JMP     U7              ; SET_CURSOR
                        4753
                        4754    ;------ CARRIAGE RETURN FOUND
                        4755
F78F                    4756    U9:
F78F B200               4757            MOV     DL,0            ; MOVE TO FIRST COLUMN
F791 EBEF               4758            JMP     U7              ; SET_CURSOR
                        4759
                        4760    ;------ LINE FEED FOUND
                        4761
F793                    4762    U10:
F793 80FE18             4763            CMP     DH,24           ; BOTTOM OF SCREEN
F796 75E8               4764            JNE     U6              ; YES, SCROLL THE SCREEN
F798 EB89               4765            JMP     U1              ; NO, JUST SET THE CURSOR
                        4766
                        4767    ;------ BELL FOUND
                        4768
F79A                    4769    U11:
F79A B302               4770            MOV     BL,2            ; SET UP COUNT FOR BEEP
F79C E8C7EE             4771            CALL    BEEP            ; SOUND THE POD BELL
F79F EB08               4772            JMP     U5              ; TTY_RETURN
                        4773    WRITE_TTY       ENDP
                        4774    ;----------------------------------------
                        4775    ; LIGHT PEN
                        4776    ;       THIS ROUTINE TESTS THE LIGHT PEN SWITCH AND THE LIGHT
                        4777    ;         PEN TRIGGER.  IF BOTH ARE SET, THE LOCATION OF THE LIGHT
                        4778    ;         PEN IS DETERMINED.  OTHERWISE, A RETURN WITH NO INFORMATION
                        4779    ;         IS MADE.
                        4780    ;       ON EXIT:
                        4781    ;       (AH) = 0 IF NO LIGHT PEN INFORMATION IS AVAILABLE
                        4782    ;               BX,CX,DX ARE DESTROYED
                        4783    ;       (AH) = 1 IF LIGHT PEN IS AVAILABLE
                        4784    ;               (DH,DL) = ROW,COLUMN OF CURRENT LIGHT PEN POSITION
                        4785    ;               (CH) = RASTER POSITION
                        4786    ;               (BX) = BEST GUESS AT PIXEL HORIZONTAL POSITION
                        4787    ;----------------------------------------
                        4788            ASSUME  CS:CODE,DS:DATA
                        4789    ;------ SUBTRACT_TABLE
F7A1                    4790    V1      LABEL   BYTE
F7A1 0303050503030304   4791            DB      3,3,5,5,3,3,3,4 ;
F7A9                    4792    READ_LPEN       PROC    NEAR
                        4793
                        4794    ;------ WAIT FOR LIGHT PEN TO BE DEPRESSED
                        4795
F7A9 B400               4796            MOV     AH,0                    ; SET NO LIGHT PEN RETURN CODE
F7AB 8B166300       R   4797            MOV     DX,ADDR_6845            ; GET BASE ADDRESS OF 6845
F7AF 83C206             4798            ADD     DX,6                    ; POINT TO STATUS REGISTER
F7B2 EC                 4799            IN      AL,DX                   ; GET STATUS REGISTER
F7B3 A804               4800            TEST    AL,4                    ; TEST LIGHT PEN SWITCH
F7B5 7578               4801            JNZ     V6                      ; NOT SET, RETURN
                        4802
                        4803    ;------ NOW TEST FOR LIGHT PEN TRIGGER
                        4804
F7B7 A802               4805            TEST    AL,2                    ; TEST LIGHT PEN TRIGGER
F7B9 747E               4806            JZ      V7                      ; RETURN WITHOUT RESETTING TRIGGER
                        4807
                        4808    ;------ TRIGGER HAS BEEN SET, READ THE VALUE IN
                        4809
F7BB B410               4810            MOV     AH,16           ; LIGHT PEN REGISTERS ON 6845
                        4811
                        4812    ;------ INPUT REGS POINTED TO BY AH, AND CONVERT TO ROW COLUMN IN DX
                        4813
F7BD 8B166300       R   4814            MOV     DX,ADDR_6845    ; ADDRESS REGISTER FOR 6845
```

```
LOC OBJ                 LINE    SOURCE

F7C1 8AC4               4815            MOV     AL,AH           ; REGISTER TO READ
F7C3 EE                 4816            OUT     DX,AL           ; SET IT UP
F7C4 42                 4817            INC     DX              ; DATA REGISTER
F7C5 EC                 4818            IN      AL,DX           ; GET THE VALUE
F7C6 8AE8               4819            MOV     CH,AL           ; SAVE IN CX
F7C8 4A                 4820            DEC     DX              ; ADDRESS REGISTER
F7C9 FEC4               4821            INC     AH
F7CB 8AC4               4822            MOV     AL,AH           ; SECOND DATA REGISTER
F7CD EE                 4823            OUT     DX,AL
F7CE 42                 4824            INC     DX              ; POINT TO DATA REGISTER
F7CF EC                 4825            IN      AL,DX           ; GET SECOND DATA VALUE
F7D0 8AE5               4826            MOV     AH,CH           ; AX HAS INPUT VALUE
                        4827
                        4828    ;------ AX HAS THE VALUE READ IN FROM THE 6845
                        4829
F7D2 8A1E4900       R   4830            MOV     BL,CRT_MODE
F7D6 2AFF               4831            SUB     BH,BH           ; MODE VALUE TO BX
F7D8 2E8A9FA1F7     R   4832            MOV     BL,CS:V1[BX]    ; DETERMINE AMOUNT TO SUBTRACT
F7DD 2BC3               4833            SUB     AX,BX           ; TAKE IT AWAY
F7DF 2B064E00       R   4834            SUB     AX,CRT_START    ; CONVERT TO CORRECT PAGE ORIGIN
F7E3 7903               4835            JNS     V2              ; IF POSITIVE, DETERMINE MODE
F7E5 B80000             4836            MOV     AX,0            ; <0 PLAYS AS '
                        4837
                        4838    ;------ DETERMINE MODE OF OPERATION
                        4839
F7E8                    4840    V2:                             ; DETERMINE_MODE
F7E8 B103               4841            MOV     CL,3            ; SET #8 SHIFT COUNT
F7EA 803E490004     R   4842            CMP     CRT_MODE,4      ; DETERMINE IF GRAPHICS OR ALPHA
F7EF 722A               4843            JB      V4              ; ALPHA_PEN
F7F1 803E490007     R   4844            CMP     CRT_MODE,7
F7F6 7423               4845            JE      V4              ; ALPHA_PEN
                        4846
                        4847    ;------ GRAPHICS MODE
                        4848
F7F8 B228               4849            MOV     DL,40           ; DIVISOR FOR GRAPHICS
F7FA F6F2               4850            DIV     DL              ; DETERMINE ROW(AL) AND COLUMN(AH)
                        4851                                    ;  AL RANGE 0-99, AH RANGE 0-39
                        4852    ;------ DETERMINE GRAPHIC ROW POSITION
                        4853
F7FC 8AE8               4854            MOV     CH,AL           ; SAVE ROW VALUE IN CH
F7FE 02ED               4855            ADD     CH,CH           ; *2 FOR EVEN/ODD FIELD
F800 8ADC               4856            MOV     BL,AH           ; COLUMN VALUE TO BX
F802 2AFF               4857            SUB     BH,BH           ; MULTIPLY BY 8 FOR MEDIUM RES
F804 803E490006     R   4858            CMP     CRT_MODE,6      ; DETERMINE MEDIUM OR HIGH RES
F809 7504               4859            JNE     V3              ; NOT_HIGH_RES
F80B B104               4860            MOV     CL,4            ; SHIFT VALUE FOR HIGH RES
F80D D0E4               4861            SAL     AH,1            ; COLUMN VALUE TIMES 2 FOR HIGH RES
F80F                    4862                                    ; NOT_HIGH_RES
F80F D3E3               4863    V3:     SHL     BX,CL           ; MULTIPLY *16 FOR HIGH RES
                        4864
                        4865    ;------ DETERMINE ALPHA CHAR POSITION
                        4866
F811 8AD4               4867            MOV     DL,AH           ; COLUMN VALUE FOR RETURN
F813 8AF0               4868            MOV     DH,AL           ; ROW VALUE
F815 D0EE               4869            SHR     DH,1            ; DIVIDE BY 4
F817 D0EE               4870            SHR     DH,1            ;  FOR VALUE IN 0-24 RANGE
F819 EB1Z               4871            JMP     SHORT V5        ; LIGHT_PEN_RETURN_SET
                        4872
                        4873    ;------ ALPHA MODE ON LIGHT PEN
                        4874
F81B                    4875    V4:                             ; ALPHA_PEN
F81B F6364A00       R   4876            DIV     BYTE PTR CRT_COLS       ; DETERMINE ROW,COLUMN VALUE
F81F 8AF0               4877            MOV     DH,AL           ; ROWS TO DH
F821 8AD4               4878            MOV     DL,AH           ; COLS TO DL
F823 D2E0               4879            SAL     AL,CL           ; MULTIPLY ROWS * 8
F825 8AE8               4880            MOV     CH,AL           ; GET RASTER VALUE TO RETURN REG
F827 8ADC               4881            MOV     BL,AH           ; COLUMN VALUE
F829 32FF               4882            XOR     BH,BH           ;  TO BX
F82B D3E3               4883            SAL     BX,CL
F820                    4884    V5:                             ; LIGHT_PEN_RETURN_SET
F820 B401               4885            MOV     AH,1            ; INDICATE EVERTHING SET
F82F                    4886    V6:                             ; LIGHT_PEN_RETURN
F82F 52                 4887            PUSH    DX              ; SAVE RETURN VALUE (IN CASE)
F830 8B166300       R   4888            MOV     DX,ADDR_6845    ; GET BASE ADDRESS
F834 83C207             4889            ADD     DX,7            ; POINT TO RESET PARM
F837 EE                 4890            OUT     DX,AL           ; ADDRESS, NOT DATA, IS IMPORTANT
```



```
LOC OBJ                 LINE    SOURCE

F838 5A                 4891            POP     DX              ; RECOVER VALUE
F839                    4892    V7:                             ; RETURN_NO_RESET
F839 5F                 4893            POP     DI
F83A 5E                 4894            POP     SI
F83B 1F                 4895            POP     DS                      ; DISCARD SAVED BX,CX,DX
F83C 1F                 4896            POP     DS
F83D 1F                 4897            POP     DS
F83E 1F                 4898            POP     DS
F83F 07                 4899            POP     ES
F840 CF                 4900            IRET
                        4901    READ_LPEN       ENDP
                        4902    ;--- INT 12 --------------------------------
                        4903    ; MEMORY_SIZE_DETERMINE
                        4904    ;       THIS ROUTINE DETERMINES THE AMOUNT OF MEMORY IN THE SYSTEM
                        4905    ;       AS REPRESENTED BY THE SWITCHES ON THE PLANAR.  NOTE THAT
                        4906    ;       THE SYSTEM MAY NOT BE ABLE TO USE I/O MEMORY UNLESS THERE
                        4907    ;       IS A FULL COMPLEMENT OF 64K BYTES ON THE PLANAR.
                        4908    ; INPUT
                        4909    ;       NO REGISTERS
                        4910    ;       THE MEMORY_SIZE VARIABLE IS SET DURING POWER ON DIAGNOSTICS
                        4911    ;        ACCORDING TO THE FOLLOWING HARDWARE ASSUMPTIONS:
                        4912    ;       PORT 60 BITS 3,2 = 00 - 16K BASE RAM
                        4913    ;                          01 - 32K BASE RAM
                        4914    ;                          10 - 48K BASE RAM
                        4915    ;                          11 - 64K BASE RAM
                        4916    ;       PORT 62 BITS 3-0 INDICATE AMOUNT OF I/O RAM IN 32K INCREMENTS
                        4917    ;               E.G., 0000 - NO RAM IN I/O CHANNEL
                        4918    ;                     0010 - 64K RAM IN I/O CHANNEL, ETC.
                        4919    ; OUTPUT
                        4920    ;       (AX) = NUMBER OF CONTIGUOUS 1K BLOCKS OF MEMORY
                        4921    ;---------------------------------------------
                        4922            ASSUME  CS:CODE,DS:DATA
F841                    4923    MEMORY_SIZE_DETERMINE   PROC    FAR
F841 FB                 4924            STI                     ; INTERRUPTS BACK ON
F842 1E                 4925            PUSH    DS              ; SAVE SEGMENT
F843 B84000         R   4926            MOV     AX,DATA         ; ESTABLISH ADDRESSING
F846 8ED8               4927            MOV     DS,AX
F848 A11300         R   4928            MOV     AX,MEMORY_SIZE  ; GET VALUE
F84B 1F                 4929            POP     DS              ; RECOVER SEGMENT
F84C CF                 4930            IRET                    ; RETURN TO CALLER
                        4931    MEMORY_SIZE_DETERMINE   ENDP
                        4932    ;--- INT 11 --------------------------------
                        4933    ; EQUIPMENT DETERMINATION
                        4934    ;       THIS ROUTINE ATTEMPTS TO DETERMINE WHAT OPTIONAL
                        4935    ;       DEVICES ARE ATTACHED TO THE SYSTEM.
                        4936    ; INPUT
                        4937    ;       NO REGISTERS
                        4938    ;       THE EQUIP_FLAG VARIABLE IS SET DURING THE POWER ON DIAGNOSTICS
                        4939    ;        USING THE FOLLOWING HARDWARE ASSUMPTIONS:
                        4940    ;       PORT 60  = LOW ORDER BITE OF EQUIPMENT
                        4941    ;       PORT 3FA = INTERRUPT ID REGISTER OF 8250
                        4942    ;                BITS 7-3 ARE ALWAYS 0
                        4943    ;       PORT 378 = OUTPUT PORT OF PRINTER -- 8255 PORT THAT
                        4944    ;                CAN BE READ AS WELL AS WRITTEN
                        4945    ; OUTPUT
                        4946    ;       (AX) IS SET, BIT SIGNIFICANT, TO INDICATE ATTACHED I/O
                        4947    ;       BIT 15,14 = NUMBER OF PRINTERS ATTACHED
                        4948    ;       BIT 13 NOT USED
                        4949    ;       BIT 12 = GAME I/O ATTACHED
                        4950    ;       BIT 11,10,9 = NUMBER OF RS232 CARDS ATTACHED
                        4951    ;       BIT 8 UNUSED
                        4952    ;       BIT 7,6 = NUMBER OF DISKETTE DRIVES
                        4953    ;                 00=1, 01=2, 10=3, 11=4 ONLY IF BIT 0 = 1
                        4954    ;       BIT 5,4 = INITIAL VIDEO MODE
                        4955    ;                 00 - UNUSED
                        4956    ;                 01 - 40X25 BW USING COLOR CARD
                        4957    ;                 10 - 80X25 BW USING COLOR CARD
                        4958    ;                 11 - 80X25 BW USING BW CARD
                        4959    ;       BIT 3,2 = PLANAR RAM SIZE (00=16K,01=32K,10=48K,11=64K)
                        4960    ;       BIT 1 NOT USED
                        4961    ;       BIT 0 = IPL FROM DISKETTE -- THIS BIT INDICATES THAT THERE ARE DISKETTE
                        4962    ;                DRIVES ON THE SYSTEM
                        4963    ;
                        4964    ;       NO OTHER REGISTERS AFFECTED
                        4965    ;---------------------------------------------
                        4966            ASSUME  CS:CODE,DS:DATA
```

```
LOC OBJ                 LINE    SOURCE

F840                    4967    EQUIPMENT       PROC    FAR
F840 FB                 4968            STI                     ; INTERRUPTS BACK ON
F84E 1E                 4969            PUSH    DS              ; SAVE SEGMENT REGISTER
F84F B84000         R   4970            MOV     AX,DATA         ; ESTABLISH ADDRESSING
F852 8ED8               4971            MOV     DS,AX
F854 A11000         R   4972            MOV     AX,EQUIP_FLAG   ; GET THE CURRENT SETTINGS
F857 1F                 4973            POP     DS              ; RECOVER SEGMENT
F858 CF                 4974            IRET                    ; RETURN TO CALLER
                        4975    EQUIPMENT       ENDP
                        4976    ;--- INT 15 ----------------------------------
                        4977    ; CASSETTE I/O
                        4978    ;       (AH) = 0  TURN CASSETTE MOTOR ON
                        4979    ;       (AH) = 1  TURN CASSETTE MOTOR OFF
                        4980    ;       (AH) = 2  READ 1 OR MORE 256 BTTE BLOCKS FROM CASSETTE
                        4981    ;                 (ES,BX) = POINTER TO DATA BUFFER
                        4982    ;                 (CX) = COUNT OF BYTES TO READ
                        4983    ;                 ON EXIT:
                        4984    ;                 (ES,BX) = POINTER TO LAST BTTE READ + 1
                        4985    ;                 (DX) = COUNT OF BYTES ACTUALLT READ
                        4986    ;                 (CY) = 0 IF NO ERROR OCCURRED
                        4987    ;                      = 1 IF ERROR OCCURRED
                        4988    ;                 (AH) = ERROR RETURN IF (CY)= 1
                        4989    ;                        = 01 IF CRC ERROR WAS DETECTED
                        4990    ;                        = 02 IF DATA TRANSITIONS ARE LOST
                        4991    ;                        = 04 IF NO DATA WAS FOUND
                        4992    ;       (AH) = 3 WRITE 1 OR MORE 256 BTTE BLOCKS TO CASSETTE
                        4993    ;                 (ES,BX) = POINTER TO DATA BUFFER
                        4994    ;                 (CX) = COUNT OF BYTES TO WRITE
                        4995    ;                 ON EXIT:
                        4996    ;                 (EX,BX) = POINTER TO LAST BTTE WRITTEN + 1
                        4997    ;                 (CX) = 0
                        4998    ;       (AH) = ANY OTHER THAN ABOVE VALUES CAUSES (CT)= 1
                        4999    ;                 AND (AH)= 80 TO BE RETURNED (INVALID COMMAND).
                        5000    ;-----------------------------------------
                        5001            ASSUME  DS:DATA, ES:NOTHING,SS:NOTHING,CS:CODE
F859                    5002    CASSETTE_IO     PROC    FAR
F859 FB                 5003            STI                             ; INTERRUPTS BACK ON
F85A 1E                 5004            PUSH    DS                      ; ESTABLISH ADDRESSING TO DATA
F85B 50                 5005            PUSH    AX
F85C B84000         R   5006            MOV     AX, DATA
F85F 8ED8               5007            MOV     DS, AX
F861 802671007F     R   5008            AND     BIOS_BREAK, 7FH         ; MAKE SURE BREAK FLAG IS OFF
F866 58                 5009            POP     AX
F867 E80400             5010            CALL    W1                      ; CASSETTE_IO_CONT
F86A 1F                 5011            POP     DS
F86B CA0200             5012            RET     2               ; INTERRUPT RETURN
                        5013    CASSETTE_IO     ENDP
F86E                    5014    W1      PROC    NEAR
                        5015    ;-----------------------------------------
                        5016    ; PURPOSE:
                        5017    ;  TO CALL APPROPRIATE ROUTINE DEPENDING ON REG AH
                        5018    ;
                        5019    ;  AH           ROUTINE
                        5020    ;_____________________________
                        5021    ;  0            MOTOR ON
                        5022    ;  1            MOTOR OFF
                        5023    ;  2            READ CASSETTE BLOCK
                        5024    ;  3            WRITE CASSETTE BLOCK
                        5025    ;_____________________________
                        5026
F86E 0AE4               5027            OR      AH,AH           ;TURN ON MOTOR?
F870 7413               5028            JZ      MOTOR_ON        ;YES, DO IT
F872 FECC               5029            DEC     AH              ;TURN OFF MOTOR?
F874 7418               5030            JZ      MOTOR_OFF       ;YES, DO IT
F876 FECC               5031            DEC     AH              ;READ CASSETTE BLOCK?
F878 741A               5032            JZ      READ_BLOCK      ;YES, DO IT
F87A FECC               5033            DEC     AH              ;WRITE CASSETTE BLOCK?
F87C 7503               5034            JNZ     W2              ; NOT_DEFINED
F87E E92701             5035            JMP     WRITE_BLOCK     ;YES, DO IT
                        5036
F881                    5037    W2:                             ;COMMAND NOT DEFINED
F881 B480               5038            MOV     AH,080H         ;ERROR, UNDEFINED OPERATION
F883 F9                 5039            STC                     ;ERROR FLAG
F884 C3                 5040            RET
                        5041    W1      ENDP
                        5042
```



```
LOC OBJ                 LINE    SOURCE

F885                    5043    MOTOR_ON        PROC    NEAR
                        5044    ;-----------------------------------
                        5045    ; PURPOSE:
                        5046    ;  TO TURN ON CASSETTE MOTOR
                        5047    ;-----------------------------------
F885 E461               5048            IN      AL,PORT_B               ;READ CASSETTE OUTPUT
F887 24F7               5049            AND     AL,NOT 08H              ; CLEAR BIT TO TURN ON MOTOR
F889 E661               5050    W3:     OUT     PORT_B,AL               ;WRITE IT OUT
F88B 2AE4               5051            SUB     AH,AH                   ;CLEAR AH
F88D C3                 5052            RET
                        5053    MOTOR_ON        ENDP
                        5054
F88E                    5055    MOTOR_OFF       PROC    NEAR
                        5056    ;-----------------------------------
                        5057    ; PURPOSE:
                        5058    ;  TO TURN CASSETTE MOTOR OFF
                        5059    ;-----------------------------------
F88E E461               5060            IN      AL,PORT_B               ;READ CASSETTE OUTPUT
F890 0C08               5061            OR      AL,08H                  ; SET BIT TO TURN OFF
F892 EBF5               5062            JMP     W3                      ;WRITE IT, CLEAR ERROR, RETURN
                        5063    MOTOR_OFF       ENDP
F894                    5064    READ_BLOCK      PROC    NEAR
                        5065    ;-----------------------------------------
                        5066    ; PURPOSE:
                        5067    ;  TO READ 1 OR MORE 256 BTTE BLOCKS FROM CASSETTE
                        5068    ;
                        5069    ; ON ENTRY:
                        5070    ;  ES IS SEGMENT FOR MEMORY BUFFER (FOR COMPACT CODE)
                        5071    ;  BX POINTS TO START OF MEMORT BUFFER
                        5072    ;  CX CONTAINS NUMBER OF BTTES TO READ
                        5073    ; ON EXIT:
                        5074    ;  BX POINTS 1 BTTE PAST LAST BTTE PUT IN MEM
                        5075    ;  CX CONTAINS DECREMENTED BYTE COUNT
                        5076    ;  DX CONTAINS NUMBER OF BTTES ACTUALLY READ
                        5077    ;
                        5078    ;  CARRT FLAG IS CLEAR IF NO ERROR DETECTED
                        5079    ;  CARRT FLAG IS SET IF CRC ERROR DETECTED
                        5080    ;-----------------------------------------
F894 53                 5081            PUSH    BX                      ;SAVE BX
F895 51                 5082            PUSH    CX                      ;SAVE CX
F896 56                 5083            PUSH    SI                      ; SAVE SI
F897 BE0700             5084            MOV     SI, 7                   ; SET UP RETRT COUNT FOR LEADER
F89A E8C201             5085            CALL    BEGIN_OP                ; BEGIN BY STARTING MOTOR
F89D                    5086    W4:                                     ; SEARCH FOR LEADER
F89D E462               5087            IN      AL,PORT_C               ;GET INTIAL VALUE
F89F 2410               5088            AND     AL,010N                 ;MASK OFF EXTRANEOUS BITS
F8A1 A26B00         R   5089            MOV     LAST_VAL,AL             ;SAVE IN LOC LAST_VAL
F8A4 BA7A3F             5090            MOV     DX,16250                ; 8 OF TRANSITIONS TO LOOK FOR
                        5091
F8A7                    5092    W5:                                     ; WAIT_FOR_EDGE
F8A7 F606710080     R   5093            TEST    BIOS_BREAK, 80H         ; CHECK FOR BREAK KEY
F8AC 7403               5094            JZ      W6                      ; JUMP IF NO BREAK KET
F8AE E98A00             5095            JMP     W17                     ; JUMP IF BREAK KEY HIT
                        5096
F8B1 4A                 5097    W6:     DEC     DX
F8B2 7503               5098            JNZ     W7                      ; JUMP IF BEGINNING OF LEADER
F8B4 E98400             5099            JMP     W17                     ; JUMP IF NO LEADER FOUND
                        5100
F8B7 E8C600             5101    W7:     CALL    READ_HALF_BIT           ;IGNORE FIRST EDGE
F8BA E3E8               5102            JCXZ    W5                      ; JUMP IF NO EDGE DETECTED
F8BC BA7803             5103            MOV     DX,0378H                ; CHECK FOR HALF BITS
F8BF B90002             5104            MOV     CX,200H                 ;MUST HAVE AT LEAST THIS MANY ONE SIZE
                        5105                                            ;PULSES BEFORE CHECKNG FOR SYNC BIT (0)
F8C2 E421               5106            IN      AL, 021H                ; INTERRUPT MASK REGISTER
F8C4 0C01               5107            OR      AL,1                    ; DISABLE TIMER INTERRUPTS
F8C6 E621               5108            OUT     021H, AL
F8C8                    5109    W8:                                     ; SEARCH-LDR
F8C8 F606710080     R   5110            TEST    BIOS_BREAK, 80H         ; CHECK FOR BREAK KEY
F8CD 756C               5111            JNZ     W17                     ; JUMP IF BREAK KEY HIT
F8CF 51                 5112            PUSH    CX                      ;SAVE REG CX
F8D0 E8AD00             5113            CALL    READ_HALF_BIT           ;GET PULSE WIDTH
F8D3 0BC9               5114            OR      CX, CX                  ; CHECK FOR TRANSITION
F8D5 59                 5115            POP     CX                      ;RESTORE ONE BIT COUNTER
F8D6 74C5               5116            JZ      W4                      ; JUMP IF NO TRANSITION
F8D8 3BD3               5117            CMP     DX,BX                   ;CHECK PULSE WIDTH
```

```
LOC OBJ                 LINE    SOURCE

F8DA E304               5118            JCXZ    W9                      ;IF CX=0 THEN WE CAN LOOK
                        5119                                            ;FOR SYNC BIT (0)
F8DC 75BF               5120            JNC     W4                      ; JUMP IF ZERO BIT (NOT GOOD LEADER)
F8DE E2E8               5121            LOOP    W8                      ;DEC CX AND READ ANOTHER HALF ONE BIT
F8E0                    5122    W9:                                     ; FIND-SYNC
F8E0 72E6               5123            JC      W8                      ; JUMP IF ONE BIT (STILL LEADER)
                        5124
                        5125    ; A SYNCH BIT HAS BEEN FOUND.  READ SYN CHARACTER:
                        5126
F8E2 E89B00             5127            CALL    READ_HALF_BIT           ;SKIP OTHER HALF OF SYNC BIT (0)
F8E5 E86A00             5128            CALL    READ_BYTE               ; READ SYN BYTE
F8E8 3C16               5129            CMP     AL, 16H                 ; SYNCHRONIZATION CHARACTER
F8EA 7549               5130            JNE     W16                     ; JUMP IF BAD LEADER FOUND.
                        5131
                        5132    ;------ GOOD CRC SO READ DATA BLOCK(S)
F8EC 5E                 5133            POP     SI                      ; RESTORE REGS
F8ED 59                 5134            POP     CX
F8EE 5B                 5135            POP     BX
                        5136    ;--------------------------------------------
                        5137    ;  READ 1 OR MORE 256 BYTE BLOCKS FROM CASSETTE
                        5138    ;
                        5139    ; ON ENTRY:
                        5140    ;  ES IS SEGMENT FOR MEMORY BUFFER (FOR COMPAC7 CODE)
                        5141    ;  BX POINTS TO START OF MEMORY BUFFER
                        5142    ;  CX CONTAINS NUMBER OF BYTES TO READ
                        5143    ; ON EXIT:
                        5144    ;  BX POINTS 1 BYTE PAST LAST BYTE PUT IN MEM
                        5145    ;  CX CONTAINS DECREMENTED BYTE COUNT
                        5146    ;  DX CONTAINS NUMBER OF BYTES ACTUALLY READ
                        5147    ;--------------------------------------------
F8EF 51                 5148            PUSH    CX                      ;SAVE BYTE COUNT
F8F0                    5149    W10:                                    ;COME HERE BEFORE EACH
                        5150                                            ;256 BYTE BLOCK IS READ
F8F0 C7066900FFFF   R   5151            MOV     CRC_REG,0FFFFH          ;INIT CRC REG
F8F6 BA0001             5152            MOV     DX,256                  ;SET DX TO DATA BLOCK SIZE
F8F9                    5153    W11:                                    ; RD_BLK
F8F9 F606710080     R   5154            TEST    BIOS_BREAK, 80H         ; CHECK FOR BREAK KEY
F8FE 7523               5155            JNZ     W13                     ; JUMP IF BREAK KEY HIT
F900 E84F00             5156            CALL    READ_BYTE               ;READ BYTE FROM CASSETTE
F903 721E               5157            JC      W13                     ;CY SET INDICATES NO DATA TRANSITIONS
F905 E305               5158            JCXZ    W12                     ;IF WE'VE ALREADY REACHED
                        5159                                            ;END OF MEMORY BUFFER
                        5160                                            ;SKIP REST OF BLOCK
F907 268807             5161            MOV     ES:[BX],AL              ;STORE DATA BYTE AT BYTE PTR
F90A 43                 5162            INC     BX                      ;INC BUFFER PTR
F90B 49                 5163            DEC     CX                      ;DEC BYTE COUNTER
F90C                    5164    W12:                    ; LOOP UNTIL DATA BLOCK HAS BEEN READ FROM CASSETTE.
F90C 4A                 5165            DEC     DX                      ;DEC BLOCK CNT
F90D 7FEA               5166            JG      W11                     ; RD_BLK
F90F E84000             5167            CALL    READ_BYTE               ;NOW READ TWO CRC BYTES
F912 E83000             5168            CALL    READ_BYTE
F915 2AE4               5169            SUB     AH,AH                   ;CLEAR AH
F917 813E69000F1D   R   5170            CMP     CRC_REG,1D0FH           ;IS THE CRC CORRECT
F91D 7506               5171            JNE     W14                     ;IF NOT EQUAL CRC IS BAD
F91F E306               5172            JCXZ    W15                     ;IF BYTE COUNT IS ZERO
                        5173                                            ;THEN WE HAVE READ ENOUGH
                        5174                                            ;SO WE WILL EXIT
F921 EBC0               5175            JMP     W10                     ;STILL MORE, SO READ ANOTHER BLOCK
F923                    5176    W13:                                    ;MISSING-DATA
                        5177                                            ;NO DATA TRANSITIONS SO
F923 B401               5178            MOV     AH,01H                  ;SET AH=02 TO INDICATE
                        5179                                            ;DATA TIMEOUT
F925                    5180    W14:                                    ; BAD-CRC
F925 FEC4               5181            INC     AH                      ;EXIT EARLY ON ERROR
                        5182                                            ;SET AH=01 TO INDICATE CRC ERROR
F927                    5183    W15:                                    ; RD-BLK-EX
F927 5A                 5184            POP     DX                      ;CALCULATE COUNT OF
F928 2BD1               5185            SUB     DX,CX                   ;DATA BYTES ACTUALLY READ
                        5186                                            ;RETURN COUNT IN REG DX
F92A 50                 5187            PUSH    AX                      ;SAVE AX (RET CODE)
F92B F6C403             5188            TEST    AH, 03H                 ; CHECK FOR ERRORS
F92E 7513               5189            JNZ     W18                     ; JUMP IF ERROR DETECTED
F930 E81F00             5190            CALL    READ_BYTE               ;READ TRAILER
F933 EB0E               5191            JMP     SHORT W18               ;SKIP TO TURN OFF MOTOR
F935                    5192    W16:                                    ; BAD-LEADER
F935 4E                 5193            DEC     SI                      ; CHECK RETRIES
```



```
LOC OBJ                 LINE    SOURCE

F936 7403               5194            JZ      W17                     ; JUMP IF TOO MANY RETRIES
F938 E962FF             5195            JMP     W4                      ; JUMP IF NOT TOO MANT RETRIES
F938                    5196    W17:                                    ; NO VALID DATA FOUND
                        5197    ;------ NO DATA FROM CASSETTE ERROR, I.E. TIMEOUT
                        5198
F938 5E                 5199            POP     SI                      ; RESTORE REGS
F939 59                 5200            POP     CX                      ;RESTORE REGS
F93A 5B                 5201            POP     BX
F93B 2BD2               5202            SUB     DX,DX                   ;ZERO NUMBER OF BYTES READ
F93D B404               5203            MOV     AH,04H                  ;TIME OUT ERROR (NO LEADER)
F93F 50                 5204            PUSH    AX
F940                    5205    W18:                                    ; MOT-OFF
F940 E421               5206            IN      AL, 021H                ; RE_ENABLE INTERRUPTS
F942 24FE               5207            AND     AL, 0FFH- 1
F944 E621               5208            OUT     021H, AL
F946 E842FF             5209            CALL    MOTOR_OFF               ;TURN OFF MOTOR
F949 58                 5210            POP     AX                      ;RESTORE RETURN CODE
F94A 80FC01             5211            CMP     AH,01H                  ;SET CARRY IF ERROR (AH>0)
F94D F5                 5212            CMC
F94E C3                 5213            REI                             ;FINISHED
                        5214    READ_BLOCK      ENDP
                        5215    ;--------------------------------------------
F94F                    5216    READ_BYTE       PROC    NEAR
                        5217    ; PURPOSE:
                        5218    ;  TO READ A BYTE FROM CASSETTE
                        5219    ;
                        5220    ; ON EXIT REG AL CONTAINS READ DATA BYTE
                        5221    ;--------------------------------------
F94F 53                 5222            PUSH    BX                      ;SAVE REGS BX,CX
F950 51                 5223            PUSH    CX
F951 B108               5224            MOV     CL,8H                   ; SET BIT COUNTER FOR 8 BITS
F953                    5225    W19:                                    ; BYTE-ASM
F953 51                 5226            PUSH    CX                      ;SAVE CX
                        5227    ;--------------------------------------
                        5228    ;  READ DATA BIT FROM CASSETTE
                        5229    ;--------------------------------------
F954 E82600             5230            CALL    READ_HALF_BIT           ;READ ONE PULSE
F957 7220               5231            JCXZ    W21                     ;IF CX=0 THEN TIMEOUT
                        5232                                            ;BECAUSE OF NO DATA TRANSITIONS
F959 53                 5233            PUSH    BX                      ;SAVE 1ST HALF BIT'S
                        5234                                            ;PULSE WIDTH (IN BX)
F95A E820               5235            CALL    READ_HALF_BIT           ;READ COMPLEMENTART PULSE
F95C 58                 5236            POP     AX                      ;COMPUTE DATA BIT
F95D 7214               5237            JCXZ    W21                     ;IF CX=0 THEN TIMEOUT DUE TO
                        5238                                            ;NO DATA TRANSITIONS
F95F 03DB               5239            ADD     BX,AX                   ;PERIOD
F961 81FBF006           5240            CMP     BX, 06F0H               ; CHECK FOR ZERO BIT
F965 F5                 5241            CMC                             ; CARRY IS SET IF ONE BIT
F966 9F                 5242            LAHF                            ;SAVE CARRY IN AH
F967 59                 5243            POP     CX                      ;RESTORE CX
                        5244                                            ;NOTE:
                        5245                                            ; MS BIT OF BYTE IS READ FIRST.
                        5246                                            ; REG CH IS SHIFTED LEFT WITH
                        5247                                            ; CARRY BEING INSERTED INTO LS
                        5248                                            ;   BIT OF CH.
                        5249                                            ; AFTER ALL 8 BITS HAVE BEEN
                        5250                                            ;   READ, THE MS BIT OF THE DATA BYTE
                        5251                                            ;   WILL BE IN THE MS BIT OF REG CH
F968 D1D5               5252            RCL     CH,1                    ;ROTATE REG CH LEFT WITH CARRY TO
                        5253                                            ;   LS BIT OF REG CH
F96A 9E                 5254            SAHF                            ;RESTORE CARRY FOR CRC ROUTINE
F96B E8D900             5255            CALL    CRC_GEN                 ;GENERATE CRC FOR BIT
F96E FEC9               5256            DEC     CL                      ;LOOP TILL ALL 8 BITS OF DATA
                        5257                                            ;ASSEMBLED IN REG CH
F970 75E0               5258            JNZ     W19                     ; BYTE_ASM
F972 8AC5               5259            MOV     AL,CH                   ;RETURN DATA BYTE IN REG AL
F974 F8                 5260            CLC
F975                    5261    W20:                                    ; RD-BYT-EX
F975 59                 5262            POP     CX                      ;RESTORE REGS CX,BX
F976 5B                 5263            POP     BX
F977 C3                 5264            RET                             ;FINISHED
F978                    5265    W21:                                    ; NO-DATA
F978 59                 5266            POP     CX                      ;RESTORE CX
F979 F9                 5267            STC                             ;INDICATE ERROR
```

```
LOC OBJ                 LINE    SOURCE

F97E EBF9               5268            JMP     W20                     ; RD_BYT_EX
                        5269    READ_BYTE       ENDP
                        5270    ;--------------------------------------
F980                    5271    READ_HALF_BIT   PROC    NEAR
                        5272    ; PURPOSE:
                        5273    ;  TO COMPUTE TIME TILL NEXT DATA
                        5274    ;  TRANSITION (EDGE)
                        5275    ;
                        5276    ; ON ENTRY:
                        5277    ;  EDGE_CNT CONTAINS LAST EDGE COUNT
                        5278    ;
                        5279    ; ON EXIT:
                        5280    ;  AX CONTAINS OLD LAST EDGE COUNT
                        5281    ;  BX CONTAINS PULSE WIDTH (HALF BIT)
                        5282    ;--------------------------------------
F980 B96400             5283            MOV     CX, 100                 ; SET TIME TO WAIT FOR BIT
F983 8A266800     R     5284            MOV     AH,LAST_VAL             ;GET PRESENT INPUT VALUE
F987                    5285    W22:                                    ; RD-H-BIT
F987 E462               5286            IN      AL,PORT_C               ;INPUT DATA BIT
F989 2410               5287            AND     AL,010H                 ;MASK OFF EXTRANEOUS BITS
F98B 3AC4               5288            CMP     AL,AH                   ;SAME AS BEFORE?
F98D E1F8               5289            LOOPE   W22                     ;LOOP TILL IT CHANGES
F98F A26800       R     5290            MOV     LAST_VAL,AL             ;UPDATE LAST_VAL WITH NEW VALUE
F992 B000               5291            MOV     AL,0                    ;READ TIMER'S COUNTER COMMAND
F994 E643               5292            OUT     TIM_CTL,AL              ;LATCH COUNTER
F996 E440               5293            IN      AL,TIMER0               ;GET LS BYTE
F998 8AE0               5294            MOV     AH,AL                   ;SAVE IN AH
F99A E440               5295            IN      AL,TIMER0               ;GET MS BYTE
F99C 86C4               5296            XCHG    AL,AH                   ;XCHG AL,AH
F99E 8B1E6700     R     5297            MOV     BX,EDGE_CNT             ;BX GETS LAST EDGE COUNT
F9A2 2BD8               5298            SUB     BX,AX                   ;SET BX EQUAL TO HALF BIT PERIOD
F9A4 A36700       R     5299            MOV     EDGE_CNT,AX             ;UPDATE EDGE COUNT;
F9A7 C3                 5300            RET
                        5301    READ_HALF_BIT   ENDP
                        5302    ;--------------------------------------
F9A8                    5303    WRITE_BLOCK     PROC    NEAR
                        5304    ;
                        5305    ; WRITE 1 OR MORE 256 BYTE BLOCKS TO CASSETTE.
                        5306    ;       THE DATA IS PADDED TO FILL OUT THE LAST 256 BYTE BLOCK.
                        5307    ;
                        5308    ;  ON ENTRY:
                        5309    ;   BX POINTS TO MEMORY BUFFER ADDRESS
                        5310    ;   CX CONTAINS NUMBER OF BYTES TO WRITE
                        5311    ;
                        5312    ;  ON EXIT:
                        5313    ;   BX POINTS 1 BYTE PAST LAST BYTE WRITTEN TO CASSETTE
                        5314    ;   CX IS ZERO
                        5315    ;--------------------------------------
F9A8 53                 5316            PUSH    BX
F9A9 51                 5317            PUSH    CX
F9AA E461               5318            IN      AL,PORT_B               ;DISABLE SPEAKER
F9AC 24FD               5319            AND     AL,NOT 02H
F9AE 0C01               5320            OR      AL, 01H                 ; ENABLE TIMER
F9B0 E661               5321            OUT     PORT_B,AL
F9B2 B0B6               5322            MOV     AL,0B6H                 ; SET UP TIMER -- MODE 3 SQUARE WAVE
F9B4 E643               5323            OUT     TIM_CTL,AL
F9B6 E8A600             5324            CALL    BEGIN_OP                ; START MOTOR AND DELAY
F9B9 B8A004             5325            MOV     AX,1184                 ; SET NORMAL BIT SIZE
F9BC E88500             5326            CALL    W31                     ; SET_TIMER
F9BF B90008             5327            MOV     CX,0800H                ;SET CX FOR LEADER BYTE COUNT
F9C2                    5328    W23:                                    ; WRITE LEADER
F9C2 F9                 5329            STC                             ; WRITE ONE BITS
F9C3 E86600             5330            CALL    WRITE_BIT               ;
F9C6 E2FA               5331            LOOP    W23                     ; LOOP 'TIL LEADER IS WRITTEN
F9C8 F8                 5332            CLC                             ;WRITE SYNC BIT (0)
F9C9 E86200             5333            CALL    WRITE_BIT
F9CC 59                 5334            POP     CX                      ;RESTORE REGS CX,BX
F9CD 5B                 5335            POP     BX
F9CE B016               5336            MOV     AL, 16H                 ; WRITE SYN CHARACTER
F9D0 E84400             5337            CALL    WRITE_BYTE              ;
```



```
LOC OBJ                 LINE    SOURCE

                        5338    ;--------------------------------------
                        5339    ;  WRITE 1 OR MORE 256 BYTE BLOCKS TO CASSETTE
                        5340    ;
                        5341    ;  ON ENTRY:
                        5342    ;   BX POINTS TO MEMORY BUFFER ADDRESS
                        5343    ;   CX CONTAINS NUMBER OF BYTES TO WRITE
                        5344    ;
                        5345    ;  ON EXIT:
                        5346    ;   BX POINTS 1 BYTE PAST LAST BYTE WRITTEN TO CASSETTE
                        5347    ;   CX IS ZERO
                        5348    ;--------------------------------------
F9D3                    5349    WR_BLOCK:
F9D3 C7066900FFFF R     5350            MOV     CRC_REG,0FFFFH          ;INIT CRC
F9D9 BA0001             5351            MOV     DX,256                  ;FOR 256 BYTES
F9DC                    5352                                            ; WR-BLK
F9DC 268A07             5353    W24:    MOV     AL,ES:[BX]              ;READ BYTE FROM MEM
F9DF E83500             5354            CALL    WRITE_BYTE              ;WRITE IT TO CASSETTE
F9E2 E302               5355            JCXZ    W25                     ;UNLESS CX=0, ADVANCE PTRS & DEC COUNT
F9E4 43                 5356            INC     BX                      ;INC BUFFER POINTER
F9E5 49                 5357            DEC     CX                      ;DEC BYTE COUNTER
F9E6                    5358                                            ; SKIP-ADV
F9E6 4A                 5359    W25:    DEC     DX                      ;DEC BLOCK CNT
F9E7 7FF3               5360            JG      W24                     ;LOOP TILL 256 BYTE BLOCK
                        5361                                            ;  IS WRITTEN TO TAPE
                        5362    ;------------------ WRITE CRC -------------
                        5363    ;  WRITE 1'S COMPLEMENT OF CRC REG TO CASSETTE
                        5364    ;  WHICH IS CHECKED FOR CORRECTNESS WHEN THE BLOCK IS READ
                        5365    ;
                        5366    ; REG AX IS MODIFIED
                        5367    ;--------------------------------------
F9E9 A16900       R     5368            MOV     AX,CRC_REG              ;WRITE THE ONE'S COMPLEMENT OF THE
                        5369                                            ;  TWO BYTE CRC TO TAPE
F9EC F7D0               5370            NOT     AX                      ;FOR 1'S COMPLEMENT
F9EE 50                 5371            PUSH    AX                      ;SAVE IT
F9EF 86E0               5372            XCHG    AH,AL                   ;WRITE MS BYTE FIRST
F9F1 E82300             5373            CALL    WRITE_BYTE              ;WRITE IT
F9F4 58                 5374            POP     AX                      ;GET IT BACK
F9F5 E81F00             5375            CALL    WRITE_BYTE              ;NOW WRITE LS BYTE
F9F8 0BC9               5376            OR      CX,CX                   ;IS BYTE COUNT EXHAUSTED?
F9FA 75D7               5377            JNZ     WR_BLOCK                ;JUMP IF NOT DONE YET
F9FC 51                 5378            PUSH    CX                      ;SAVE REG CX
F9FD B92000             5379            MOV     CX, 32                  ;WRITE OUT TRAILER BITS
FA00                    5380                                            ; TRAIL-LOOP
FA00 F9                 5381    W26:    STC
FA01 E8FA00             5382            CALL    WRITE_BIT
FA04 E2FA               5383            LOOP    W26                     ; WRITE UNTIL TRAILER WRITTEN
FA06 59                 5384            POP     CX                      ;RESTORE REG CX
FA07 B080               5385            MOV     AL, 080H                ; TURN TIMER2 OFF
FA09 E643               5386            OUT     TIM_CTL, AL
FA0B B80100             5387            MOV     AX, 1
FA0E E83300             5388            CALL    W31                     ; SET_TIMER
FA11 E87AFE             5389            CALL    MOTOR_OFF               ;TURN MOTOR OFF
FA14 2BC0               5390            SUB     AX,AX                   ;NO ERRORS REPORTED ON WRITE OP
FA16 C3                 5391            RET                             ;FINISHED
                        5392    WRITE_BLOCK     ENDP
                        5393    ;--------------------------------------
FA17                    5394    WRITE_BYTE      PROC    NEAR
                        5395    ;  WRITE A BYTE TO CASSETTE.
                        5396    ;  BYTE TO WRITE IS IN REG AL.
                        5397    ;--------------------------------------
FA17 51                 5398            PUSH    CX                      ;SAVE REGS CX,AX
FA18 50                 5399            PUSH    AX
FA19 8AE8               5400            MOV     CH,AL                   ;AL=BYTE TO WRITE.
                        5401                                            ;   (MS BIT WRITTEN FIRST)
FA1B B108               5402            MOV     CL,8                    ;FOR 8 DATA BITS IN BYTE.
                        5403                                            ;   NOTE: TWO EDGES PER BIT
FA1D                    5404    W27:                                    ; DISASSEMBLE THE DATA BIT
FA1D D0D5               5405            RCL     CH,1                    ;ROTATE MS BIT INTO CARRY
FA1F 9C                 5406            PUSHF                           ;SAVE FLAGS.
                        5407                                            ;   NOTE: DATA BIT IS IN CARRY
FA20 E80B00             5408            CALL    WRITE_BIT               ;WRITE DATA BIT
FA23 90                 5409            POPF                            ;RESTORE CARRY FOR CRC CALC
FA24 E82400             5410            CALL    CRC_GEN                 ;COMPUTE CRC ON DATA BIT
FA27 FEC9               5411            DEC     CL                      ;LOOP TILL ALL 8 BITS DONE
FA29 75F2               5412            JNZ     W27                     ; JUMP IF NOT DONE YET
FA2B 58                 5413            POP     AX                      ;RESTORE REGS AX,CX
FA2C 59                 5414            POP     CX
```

```
LOC OBJ                 LINE    SOURCE

FA2D C3                 5415            RET                             ;WE ARE FINISHED
                        5416    WRITE_BYTE      ENDP
                        5417    ;---------------------------------------
FA2E                    5418    WRITE_BIT       PROC    NEAR
                        5419    ; PURPOSE:
                        5420    ;
                        5421    ;  TO WRITE A DATA BIT TO CASSETTE
                        5422    ;  CARRY FLAG CONTAINS DATA BIT
                        5423    ;  I.E. IF SET   DATA BIT IS A ONE
                        5424    ;       IF CLEAR DATA BIT IS A ZERO
                        5425    ;
                        5426    ;  NOTE: TWO EDGES ARE WRITTEN PER BIT
                        5427    ;        ONE BIT  HAS 500 USEC BETWEEN EDGES
                        5428    ;            FOR A 1000 USEC PERIOD (1 MILLISEC)
                        5429    ;
                        5430    ;        ZERO BIT HAS 250 USEC BETWEEM EDGES
                        5431    ;            FOR A  500 USEC PERIOD (.5 MILLISEC)
                        5432    ; CARRY FLAG IS DATA BIT
                        5433    ;---------------------------------------
                        5434                                            ;ASSUME IT'S A '1'
FA2E B8A004             5435            MOV     AX,1184                 ; SET AX TO NOMINAL ONE SIZE
FA31 7203               5436            JC      W28                     ; JUMP IF ONE BIT
FA33 B85002             5437            MOV     AX,592                  ; NO, SET TO NOMINAL ZERO SIZE
FA36                    5438    W28:                                    ; WRITE-BIT-AX
FA36 50                 5439            PUSH    AX                      ;WRITE BIT WITH PERIOD EQ TO VALUE AX
FA37                    5440    W29:
FA37 E462               5441            IN      AL,PORT_C               ;INPUT TIMER_0 OUTPUT
FA39 2420               5442            AND     AL,020H
FA3B 74FA               5443            JZ      W29                     ;LOOP TILL HIGH
FA3D                    5444    W30:
FA3D E462               5445            IN      AL,PORT_C               ;NOW WAIT TILL TIMER'S OUTPUT IS LOW
FA3F 2420               5446            AND     AL,020H
FA41 75FA               5447            JNZ     W30
                        5448                                            ;RELOAD TIMER WITH PERIOD
                        5449                                            ;FOR NEXT DATA BIT
FA43 58                 5450            POP     AX                      ;RESTORE PERIOD COUNT
FA44                    5451    W31:                                    ; SET TIMER
FA44 E642               5452            OUT     042H, AL                ; SET LOW BYTE OF TIMER 2
FA46 8AC4               5453            MOV     AL, AH
FA48 E642               5454            OUT     042H, AL                ; SET HIGH BYTE OF TIMER 2
FA4A C3                 5455            RET
                        5456    WRITE_BIT       ENDP
                        5457    ;---------------------------------------
FA4B                    5458    CRC_GEN         PROC    NEAR
                        5459    ;  UPDATE CRC REGISTER WITH NEXT DATA BIT
                        5460    ;
                        5461    ;  CRC IS USED TO DETECT READ ERRORS
                        5462    ;
                        5463    ;  ASSUMES DATA BIT IS IN CARRY
                        5464    ;
                        5465    ;  REG AX IS MODIFIED
                        5466    ;  FLAGS ARE MODIFIED
                        5467    ;---------------------------------------
FA4B A16900         R   5468            MOV     AX,CRC_REG
                        5469                                            ;THE FOLLOWING INSTUCTIONS
                        5470                                            ;WILL SET THE OVERFLOW FLAG
                        5471                                            ;IF CARRY AND MS BIT OF CRC
                        5472                                            ;ARE UNEQUAL
FA4E D1D8               5473            RCR     AX,1
FA50 D1D0               5474            RCL     AX,I
FA52 F8                 5475            CLC                             ;CLEAR CARRY
FA53 7104               5476            JNO     W32                     ;SKIP IF NO OVERFLOW
                        5477                                            ;IF DATA BIT XORED WITH
                        5478                                            ; CRC REG BIT 15 IS ONE
FA55 351008             5479            XOR     AX,0810H                ;THEN XOR CRC REG WITH
                        5480                                            ; 0810H
FA58 F9                 5481            STC                             ;SET CARRY
FA59                    5482    W32:
FA59 D1D0               5483            RCL     AX,1                    ;ROTATE CARRY (DATA BIT)
                        5484                                            ;INTO CRC REG
FA5B A36900         R   5485            MOV     CRC_REG,AX              ;UPDATE CRC_REG
FA5E C3                 5486            RET                             ;FINISNED
                        5487    CRC_GEN         ENDP
                        5488    ;---------------------------------------
FA5F                    5489    BEGIN_OP        PROC    NEAR            ; START TAPE AND DELAY
                        5490    ;
                        5491    ;---------------------------------------
```



```
LOC OBJ                 LINE    SOURCE

FA5F E823FE             5492            CALL    MOTOR_ON                ;TURN ON MOTOR
FA62 8342               5493            MOV     BL,42H                  ;DELAY FOR TAPE DRIVE
                        5494                                            ;TO GET UP TO SPEED  (1/2 SEC)
FA64                    5495    W33:
FA64 B90007             5496            MOV     CX,700H                 ;INNER LOOP= APPROX. 10 MILLISEC
FA67 E2FE               5497    W34:    LOOP    W34
FA69 FECB               5498            DEC     BL
FA6B 75F7               5499            JNZ     W33
FA6D C3                 5500            RET
                        5501    BEGIN_OP        ENDP
                        5502    ;---------------------------------------
                        5503    ; CHARACTER GENERATOR GRAPHICS FOR 320X200 AND 640X200 GRAPHICS
                        5504    ;---------------------------------------
FA6E                    5505    CRT_CHAR_GEN    LABEL   BYTE
FA6E 0000000000000000   5506            DB      000H,000H,000H,000H,000H,000H,000H,000H ; 0_00
FA76 7E81A581BD99817E   5507            DB      07EH,081H,0A5H,081H,0BDH,099H,081H,07EH ; 0_01
FA7E 7EFF DBFFC3E7FF7E  5508            DB      07EH,0FFH,0DBH,0FFH,0C3H,0E7H,0FFH,07EH ; 0_02
FA86 6CFEFEFE7C381000   5509            DB      06CH,0FEH,0FEH,0FEH,07CH,038H,010H,000H ; 0_03
FA8E 10387CFE7C381008   5510            DB      010H,038H,07CH,0FEH,07CH,038H,010H,000H ; 0_04
FA96 387C38FEFE7C387C   5511            DB      038H,07CH,038H,0FEH,0FEH,07CH,038H,07CH ; 0_05
FA9E 1010387CFE7C387C   5512            DB      010H,010H,038H,07CH,0FEH,07CH,038H,07CH ; 0_06
FAA6 0000183C3C180000   5513            DB      000H,000H,018H,03CH,03CH,018H,000H,000H ; 0_07
FAAE FFFFE7C3C3E7FFFF   5514            DB      0FFH,0FFH,0E7H,0C3H,0C3H,0E7H,0FFH,0FFH ; 0_08
FAB6 003C664242663C00   5515            DB      000H,03CH,066H,042H,042H,066H,03CH,000H ; 0_09
FABE FFC399BDBD99C3FF   5516            DB      0FFH,0C3H,099H,0BDH,0BDH,099H,0C3H,0FFH ; 0_0A
FAC6 0F070F7DCCCCCC78   5517            DB      00FH,007H,00FH,07DH,0CCH,0CCH,0CCH,078H ; 0_0B
FACE 3C6666663C187E18   5518            DB      03CH,066H,066H,066H,03CH,018H,07EH,018H ; 0_0C
FAD6 3F333F303070F0E0   5519            DB      03FH,033H,03FH,030H,030H,070H,0F0H,0E0H ; 0_0D
FADE 7F637F636367E6C0   5520            DB      07FH,063H,07FH,063H,063H,067H,0E6H,0C0H ; 0_0E
FAE6 995A3CE7E73C5A99   5521            DB      099H,05AH,03CH,0E7H,0E7H,03CH,05AH,099H ; 0_0F
                        5522
FAEE 80E0F8FEF8E08000   5523            DB      080H,0E0H,0F8H,0FEH,0F8H,0E0H,080H,000H ; 0_10
FAF6 020E3EFE3E0E0200   5524            DB      002H,00EH,03EH,0FEH,03EH,00EH,002H,000H ; 0_11
FAFE 183C7E18187E3C18   5525            DB      018H,03CH,07EH,018H,018H,07EH,03CH,018H ; 0_12
FB06 6666666666006600   5526            DB      066H,066H,066H,066H,066H,000H,066H,000H ; 0_13
FB0E 7FDBDB7B1B1B1B00   5527            DB      07FH,0DBH,0DBH,07BH,01BH,01BH,01BH,000H ; 0_14
FB16 3E63386C6C386CC78  5528            DB      03EH,063H,038H,06CH,06CH,038H,0CCH,078H ; 0_15
FB1E 000000007E7E7E00   5529            DB      000H,000H,000H,000H,07EH,07EH,07EH,000H ; 0_16
FB26 183C7E187E3C18FF   5530            DB      018H,03CH,07EH,018H,07EH,03CH,018H,0FFH ; 0_17
FB2E 183C7E1818181800   5531            DB      018H,03CH,07EH,018H,018H,018H,018H,000H ; 0_18
FB36 1818181187E3C1800  5532            DB      018H,018H,018H,018H,07EH,03CH,018H,000H ; 0_19
FB3E 00180CFE0C180000   5533            DB      000H,018H,00CH,0FEH,00CH,018H,000H,000H ; 0_1A
FB46 00306 0FE6030000 0 5534            DB      000H,030H,060H,0FEH,060H,030H,000H,000H ; 0_1B
FB4E 0000C0C0C0FE0000   5535            DB      000H,000H,0C0H,0C0H,0C0H,0FEH,000H,000H ; 0_1C
FB56 002466FF66240000   5536            DB      000H,024H,066H,0FFH,066H,024H,000H,000H ; 0_1D
FB5E 00183C7EFFFF0000   5537            DB      000H,018H,03CH,07EH,0FFH,0FFH,000H,000H ; 0_1E
FB66 00FFFF7E3C180000   5538            DB      000H,0FFH,0FFH,07EH,03CH,018H,000H,000H ; 0_1F
                        5539
FB6E 0000000000000000   5540            DB      000H,000H,000H,000H,000H,000H,000H,000H ; SP D_20
FB76 3078783030003000   5541            DB      030H,078H,078H,030H,030H,000H,030H,000H ; ! D_21
FB7E 6C6C6C0000000000   5542            DB      06CH,06CH,06CH,000H,000H,000H,000H,000H ; " D_22
FB86 6C6CFE6CFE6C6C00   5543            DB      06CH,06CH,0FEH,06CH,0FEH,06CH,06CH,000H ; # D_23
FB8E 307CC0780CF83000   5544            DB      030H,07CH,0C0H,078H,00CH,0F8H,030H,000H ; $ D_24
FB96 00C6CC183066C600   5545            DB      000H,0C6H,0CCH,018H,030H,066H,0C6H,000H ; PER CENT D_25
FB9E 386C3876DCCC7600   5546            DB      038H,06CH,038H,076H,0DCH,0CCH,076H,000H ; & D_26
FBA6 6060C000000000 0   5547            DB      060H,060H,0C0H,000H,000H,000H,000H,000H ; ' D_27
FBAE 183060606030180 0  5548            DB      018H,030H,060H,060H,060H,030H,018H,000H ; ( D_28
FBB6 6030181818306000   5549            DB      060H,030H,018H,018H,018H,030H,060H,000H ; ) D_29
FBBE 00663CFF3C660000   5550            DB      000H,066H,03CH,0FFH,03CH,066H,000H,000H ; * D_2A
FBC6 003030FC30300000   5551            DB      000H,030H,030H,0FCH,030H,030H,000H,000H ; + D_2B
FBCE 0000000003030360   5552            DB      000H,000H,000H,000H,000H,030H,030H,060H ; , D_2C
FBD6 000000FC00000000   5553            DB      000H,000H,000H,0FCH,000H,000H,000H,000H ; - D_2D
FBDE 0000000003030000   5554            DB      000H,000H,000H,000H,000H,030H,030H,000H ; . D_2E
FBE6 060C183060C08000   5555            DB      006H,00CH,018H,030H,060H,0C0H,080H,000H ; / D_2F
                        5556
FBEE 7CC6CEDEF6E67C00   5557            DB      07CH,0C6H,0CEH,0DEH,0F6H,0E6H,07CH,000H ; 0 D_30
FBF6 307030303030FC00   5558            DB      030H,070H,030H,030H,030H,030H,0FCH,000H ; 1 D_31
FBFE 78CC0C3860CCFC00   5559            DB      078H,0CCH,00CH,038H,060H,0CCH,0FCH,000H ; 2 D_32
FC06 78CC0C380CCC7800   5560            DB      078H,0CCH,00CH,038H,00CH,0CCH,078H,000H ; 3 D_33
FC0E 1C3C6CCCFE0C1E00   5561            DB      01CH,03CH,06CH,0CCH,0FEH,00CH,01EH,000H ; 4 D_34
FC16 FCC0F80C0CCC7800   5562            DB      0FCH,0C0H,0F8H,00CH,00CH,0CCH,078H,000H ; 5 D_35
FC1E 3860C0F8CCCC7800   5563            DB      038H,060H,0C0H,0F8H,0CCH,0CCH,078H,000H ; 6 D_36
FC26 FCCC0C1830303000   5564            DB      0FCH,0CCH,00CH,018H,030H,030H,030H,000H ; 7 D_37
FC2E 78CCCC78CCCC7800   5565            DB      078H,0CCH,0CCH,078H,0CCH,0CCH,078H,000H ; 8 D_38
FC36 78CCCC7C0C187000   5566            DB      078H,0CCH,0CCH,07CH,00CH,018H,070H,000H ; 9 D_39
FC3E 0030300000300 0    5567            DB      000H,030H,030H,000H,000H,030H,030H,000H ; : D_3A
```

```
LOC OBJ                 LINE    SOURCE

FC46 0030300000303060    5568            DB      000H,030H,030H,000H,000H,030H,030H,060H ; ; D_3B
FC4E 183060C060301800    5569            DB      018H,030H,060H,0C0H,060H,030H,018H,000H ; < D_3C
FC56 0000FC0000FC0000    5570            DB      000H,000H,0FCH,000H,000H,0FCH,000H,000H ; = D_3D
FC5E 6030180C18306000    5571            DB      060H,030H,018H,00CH,018H,030H,060H,000H ; > D_3E
FC66 78CC0C1830003000    5572            DB      078H,0CCH,00CH,018H,030H,000H,030H,000H ; ? D_3F
                         5573
FC6E 7CC60DEDEDEC07800   5574            DB      07CH,0C6H,0DEH,0DEH,0DEH,0C0H,078H,000H ; @ D_40
FC76 3078CCCCFCCCCC00    5575            DB      030H,078H,0CCH,0CCH,0FCH,0CCH,0CCH,000H ; A D_41
FC7E FC66667C6666FC00    5576            DB      0FCH,066H,066H,07CH,066H,066H,0FCH,000H ; B D_42
FC86 3C66C0C0C0663C00    5577            DB      03CH,066H,0C0H,0C0H,0C0H,066H,03CH,000H ; C D_43
FC8E F86C6666666CF800    5578            DB      0F8H,06CH,066H,066H,066H,06CH,0F8H,000H ; D D_44
FC96 FE6268786862FE00    5579            DB      0FEH,062H,068H,078H,068H,062H,0FEH,000H ; E D_45
FC9E FE6268786860F000    5580            DB      0FEH,062H,068H,078H,068H,060H,0F0H,000H ; F D_46
FCA6 3C66C0C0CE663E00    5581            DB      03CH,066H,0C0H,0C0H,0CEH,066H,03EH,000H ; G D_47
FCAE CCCCCCFCCCCCCC00    5582            DB      0CCH,0CCH,0CCH,0FCH,0CCH,0CCH,0CCH,000H ; H D_48
FCB6 7830303030307800    5583            DB      078H,030H,030H,030H,030H,030H,078H,000H ; I D_49
FCBE 1E0C0C0CCCCC7800    5584            DB      01EH,00CH,00CH,00CH,0CCH,0CCH,078H,000H ; J D_4A
FCC6 E6666C786C66E600    5585            DB      0E6H,066H,06CH,078H,06CH,066H,0E6H,000H ; K D_4B
FCCE F06060606266FE00    5586            DB      0F0H,060H,060H,060H,062H,066H,0FEH,000H ; L D_4C
FCD6 C6EEFEFED6C6C600    5587            DB      0C6H,0EEH,0FEH,0FEH,0D6H,0C6H,0C6H,000H ; M D_4D
FCDE C6E6F6DECEC6C600    5588            DB      0C6H,0E6H,0F6H,0DEH,0CEH,0C6H,0C6H,000H ; N D_4E
FCE6 386CC6C6C66C3800    5589            DB      038H,06CH,0C6H,0C6H,0C6H,06CH,038H,000H ; O D_4F
                         5590
FCEE FC66667C6060F000    5591            DB      0FCH,066H,066H,07CH,060H,060H,0F0H,000H ; P D_50
FCF6 78CCCCCCDC781C00    5592            DB      078H,0CCH,0CCH,0CCH,0DCH,078H,01CH,00CH ; Q D_51
FCFE FC66667C6C66E600    5593            DB      0FCH,066H,066H,07CH,06CH,066H,0E6H,000H ; R D_52
FD06 78CCE0701CCC7800    5594            DB      078H,0CCH,0E0H,070H,01CH,0CCH,078H,000H ; S D_53
FD0E FCB4303030307800    5595            DB      0FCH,0B4H,030H,030H,030H,030H,078H,000H ; T D_54
FD16 CCCCCCCCCCCCFC00    5596            DB      0CCH,0CCH,0CCH,0CCH,0CCH,0CCH,0FCH,000H ; U D_55
FD1E CCCCCCCCCC783000    5597            DB      0CCH,0CCH,0CCH,0CCH,0CCH,078H,030H,000H ; V D_56
FD26 C6C6C6D6FEEEC600    5598            DB      0C6H,0C6H,0C6H,0D6H,0FEH,0EEH,0C6H,000H ; W D_57
FD2E C6C66C38386CC600    5599            DB      0C6H,0C6H,06CH,038H,038H,06CH,0C6H,000H ; X D_58
FD36 CCCCCC7830307800    5600            DB      0CCH,0CCH,0CCH,078H,030H,030H,078H,000H ; Y D_59
FD3E FEC68C183266FE00    5601            DB      0FEH,0C6H,08CH,018H,032H,066H,0FEH,000H ; Z D_5A
FD46 7860606060607800    5602            DB      078H,060H,060H,060H,060H,060H,078H,000H ; [ D_5B
FD4E C06030180C060200    5603            DB      0C0H,060H,030H,018H,00CH,006H,002H,000H ; BACKSLASH D_5C
FD56 7818181818187800    5604            DB      078H,018H,018H,018H,018H,018H,078H,000H ; ] D_5D
FD5E 10386CC600000000    5605            DB      010H,038H,06CH,0C6H,000H,000H,000H,000H ; CIRCUMFLEX D_5E
FD66 00000000000000FF    5606            DB      000H,000H,000H,000H,000H,000H,000H,0FFH ; _ D_5F
                         5607
FD6E 3030180000000000    5608            DB      030H,030H,018H,000H,000H,000H,000H,000H ;   D_60
FD76 0000780C7CCC7600    5609            DB      000H,000H,078H,00CH,07CH,0CCH,076H,000H ; LOWER CASE A D_61
FD7E E060607C6666DC00    5610            DB      0E0H,060H,060H,07CH,066H,066H,0DCH,000H ; L.C. B D_62
FD86 000078CCC0CC7800    5611            DB      000H,000H,078H,0CCH,0C0H,0CCH,078H,000H ; L.C. C D_63
FD8E 1C0C0C7CCCCC7600    5612            DB      01CH,00CH,00CH,07CH,0CCH,0CCH,076H,000H ; L.C. D D_64
FD96 000078CCFCC07800    5613            DB      000H,000H,078H,0CCH,0FCH,0C0H,078H,000H ; L.C. E D_65
FD9E 386C60F06060F000    5614            DB      038H,06CH,060H,0F0H,060H,060H,0F0H,000H ; L.C. F D_66
FDA6 000076CCCC7C0CF8    5615            DB      000H,000H,076H,0CCH,0CCH,07CH,00CH,0F8H ; L.C. G D_67
FDAE E0606C766666E600    5616            DB      0E0H,060H,06CH,076H,066H,066H,0E6H,000H ; L.C. H D_68
FDB6 3000703030307800    5617            DB      030H,000H,070H,030H,030H,030H,078H,000H ; L.C. I D_69
FDBE 0C000C0C0CCCCC78    5618            DB      00CH,000H,00CH,00CH,00CH,0CCH,0CCH,078H ; L.C. J D_6A
FDC6 E0606666C786CE600   5619            DB      0E0H,060H,066H,06CH,078H,06CH,0E6H,000H ; L.C. K D_6B
FDCE 7030303030307800    5620            DB      070H,030H,030H,030H,030H,030H,078H,000H ; L.C. L D_6C
FDD6 0000CCFEFED6C600    5621            DB      000H,000H,0CCH,0FEH,0FEH,0D6H,0C6H,000H ; L.C. M D_6D
FDDE 0000F8CCCCCCCC00    5622            DB      000H,000H,0F8H,0CCH,0CCH,0CCH,0CCH,000H ; L.C. N D_6E
FDE6 000078CCCCCC7800    5623            DB      000H,000H,078H,0CCH,0CCH,0CCH,078H,000H ; L.C. O D_6F
                         5624
FDEE 0000DC66667C60F0    5625            DB      000H,000H,0DCH,066H,066H,07CH,060H,0F0H ; L.C. P D_70
FDF6 000076CCCC7C0C1E    5626            DB      000H,000H,076H,0CCH,0CCH,07CH,00CH,01EH ; L.C. Q D_71
FDFE 0000DC766660F000    5627            DB      000H,000H,0DCH,076H,066H,060H,0F0H,000H ; L.C. R D_72
FE06 00007CC0780CF800    5628            DB      000H,000H,07CH,0C0H,078H,00CH,0F8H,000H ; L.C. S D_73
FE0E 10307C3030341800    5629            DB      010H,030H,07CH,030H,030H,034H,018H,000H ; L.C. T D_74
FE16 0000CCCCCCCC7600    5630            DB      000H,000H,0CCH,0CCH,0CCH,0CCH,076H,000H ; L.C. U D_75
FE1E 0000CCCCCC783000    5631            DB      000H,000H,0CCH,0CCH,0CCH,078H,030H,000H ; L.C. V D_76
FE26 0000C6D6FEFE6C00    5632            DB      000H,000H,0C6H,0D6H,0FEH,0FEH,06CH,000H ; L.C. W D_77
FE2E 0000C66C386CC600    5633            DB      000H,000H,0C6H,06CH,038H,06CH,0C6H,000H ; L.C. X D_78
FE36 0000CCCCCC7C0CF8    5634            DB      000H,000H,0CCH,0CCH,0CCH,07CH,00CH,0F8H ; L.C. Y D_79
FE3E 0000FC983064FC00    5635            DB      000H,000H,0FCH,098H,030H,064H,0FCH,000H ; L.C. Z D_7A
FE46 1C3030E030301C00    5636            DB      01CH,030H,030H,0E0H,030H,030H,01CH,000H ;   D_7B
FE4E 1818180018181800    5637            DB      018H,018H,018H,000H,018H,018H,018H,000H ;   D_7C
FE56 E030301C3030E000    5638            DB      0E0H,030H,030H,01CH,030H,030H,0E0H,000H ;   D_7D
FE5E 76DC000000000000    5639            DB      076H,0DCH,000H,000H,000H,000H,000H,000H ;   D_7E
FE66 0010386CC6C6FE00    5640            DB      000H,010H,038H,06CH,0C6H,0C6H,0FEH,000H ; DELTA D_7F
```



```
LOC OBJ                 LINE    SOURCE

                         5641    ;--- INT 1A --------------------------------
                         5642    ; TIME_OF_DAY
                         5643    ;  THIS ROUTINE ALLOWS THE CLOCK TO BE SET/READ
                         5644    ;
                         5645    ; INPUT
                         5646    ;   (AH) = 0    READ THE CURRENT CLOCK SETTING
                         5647    ;               RETURNS CX = HIGH PORTION OF COUNT
                         5648    ;                       DX = LOW PORTION OF COUNT
                         5649    ;                       AL = 0 IF TIMER HAS NOT PASSED 24 HOURS SINCE LAST READ
                         5650    ;                          <>0 IF ON ANOTHER DAY
                         5651    ;   (AH) = 1    SET THE CURRENT CLOCK
                         5652    ;       CX = HIGH PORTION OF COUNT
                         5653    ;       DX = LOW PORTION OF COUNT
                         5654    ; NOTE: COUNTS OCCUR AT THE RATE OF 1193180/65536 COUNTS/SEC
                         5655    ;       (OR ABOUT 18.2 PER SECOND -- SEE EQUATES BELOW)
                         5656    ;-----------------------------------------------
                         5657            ASSUME  CS:CODE,DS:DATA
FE6E                     5658    TIME_OF_DAY     PROC    FAR
FE6E FB                  5659            STI                     ; INTERRUPTS BACK ON
FE6F 1E                  5660            PUSH    DS              ; SAVE SEGMENT
FE70 50                  5661            PUSH    AX              ; SAVE PARM
FE71 884000          R   5662            MOV     AX,DATA
FE74 8ED8                5663            MOV     DS,AX           ; ESTABLISH ADDRESSING TO VALUES
FE76 58                  5664            POP     AX              ; GET BACK INPUT PARM
FE77 0AE4                5665            OR      AH,AH           ; AH=0
FE79 7407                5666            JZ      T2              ; READ_TIME
FE7B FECC                5667            DEC     AH              ; AH=1
FE7D 7416                5668            JZ      T3              ; SET_TIME
FE7F                     5669    T1:                             ; TOD_RETURN
FE7F FB                  5670            STI                     ; INTERRUPTS BACK ON
FE80 1F                  5671            POP     DS              ; RECOVER SEGMENT
FE81 CF                  5672            IRET                    ; RETURN TO CALLER
                         5673
FE82                     5674    T2:                             ; READ_TIME
FE82 FA                  5675            CLI                     ; NO TIMER INTERRUPTS WHILE READING
FE83 A07000          R   5676            MOV     AL,TIMER_OFL
FE86 C606700000      R   5677            MOV     TIMER_OFL,0     ; GET OVERFLOW, AND RESET THE FLAG
FE8B 8B0E6E00        R   5678            MOV     CX,TIMER_HIGH
FE8F 8B166C00        R   5679            MOV     DX,TIMER_LOW
FE93 EBEA                5680            JMP     T1                      ; TOD_RETURN
                         5681
FE95                     5682    T3:                             ; SET_TIME
FE95 FA                  5683            CLI                     ; NO INTERRUPTS WHILE WRITING
FE96 89166C00        R   5684            MOV     TIMER_LOW,DX
FE9A 890E6E00        R   5685            MOV     TIMER_HIGH,CX   ; SET THE TIME
FE9E C606700000      R   5686            MOV     TIMER_OFL,0     ; RESET OVERFLOW
FEA3 EBDA                5687            JMP     T1              ; TOD_RETURN
                         5688    TIME_OF_DAY     ENDP
                         5689    ;-----------------------------------------------
                         5690    ; THIS ROUTINE HANDLES THE TIMER INTERRUPT FROM
                         5691    ; CHANNEL 0 OF THE 8253 TIMER.  INPUT FREQUENCY IS 1.19318 MHZ
                         5692    ; AND THE DIVISOR IS 65536, RESULTING IN APPROX. 18.2 INTERRUPTS
                         5693    ; EVERY SECOND.
                         5694    ;
                         5695    ; THE INTERRUPT HANDLER MAINTAINS A COUNT OF INTERRUPTS SINCE POWER
                         5696    ;  ON TIME, WHICH MAY BE USED TO ESTABLISH TIME OF DAY.
                         5697    ; THE INTERRUPT HANDLER ALSO DECREMENTS THE MOTOR CONTROL COUNT
                         5698    ;  OF THE DISKETTE, AND WHEN IT EXPIRES, WILL TURN OFF THE DISKETTE
                         5699    ;  MOTOR, AND RESET THE MOTOR RUNNING FLAGS
                         5700    ; THE INTERRUPT HANDLER WILL ALSO INVOKE A USER ROUTINE THROUGH INTERRUPT
                         5701    ;  1CH AT EVERY TIME TICK.  THE USER MUST CODE A ROUTINE AND PLACE THE
                         5702    ;  CORRECT ADDRESS IN THE VECTOR TABLE.
                         5703    ;-----------------------------------------------
FEA5                     5704    TIMER_INT       PROC    FAR
FEA5 FB                  5705            STI                     ; INTERRUPTS BACK ON
FEA6 1E                  5706            PUSH    DS
FEA7 50                  5707            PUSH    AX
FEA8 52                  5708            PUSH    DX              ; SAVE MACHINE STATE
FEA9 884000          R   5709            MOV     AX,DATA
FEAC 8ED8                5710            MOV     DS,AX           ; ESTABLISH ADDRESSABILITY
FEAE FF066C00        R   5711            INC     TIMER_LOW       ; INCREMENT TIME
FEB2 7504                5712            JNZ     T4              ; TEST_DAY
FEB4 FF066E00        R   5713            INC     TIMER_HIGH      ; INCREMENT HIGH WORD OF TIME
FEB8                     5714    T4:                             ; TEST_DAY
```

```
LOC OBJ                 LINE    SOURCE

FEB6 833E6E0018    R    5715            CMP     TIMER_HIGH,018H ; TEST FOR COUNT EQUALLING 24 HOURS
FEBB 7519           R    5716            JNZ     T5              ; DISKETTE_CTL
FEBF 813E6C00B000   R    5717            CMP     TIMER_LOW,0B0H
FEC5 7511           R    5718            JNZ     T5              ; DISKETTE_CTL
                         5719
                         5720    ;------ TIMER HAS GONE 24 HOURS
                         5721
FEC7 C7066E000000   R    5722            MOV     TIMER_HIGH,0
FECD C7066C000000   R    5723            MOV     TIMER_LOW,0
FED3 C606700001     R    5724            MOV     TIMER_OFL,1
                         5725
                         5726    ;------ TEST FOR DISKETTE TIME OUT
                         5727
FED8                     5728    T5:                             ; DISKETTE_CTL
FED8 FE0E4000       R    5729            DEC     MOTOR_COUNT
FEDC 750B           R    5730            JNZ     T6              ; RETURN IF COUNT NOT OUT
FEDE 80263F00F0     R    5731            AND     MOTOR_STATUS,0F0H       ; TURN OFF MOTOR RUNNING BITS
FEE3 B00C           R    5732            MOV     AL,0CH
FEE5 BAF203         R    5733            MOV     DX,03F2H                ; FDC CTL PORT
FEE8 EE                  5734            OUT     DX,AL           ; TURN OFF THE MOTOR
                         5735
FEE9                     5736    T6:                             ; TIMER_RET:
FEE9 CD1C                5737            INT     1CH             ; TRANSFER CONTROL TO A USER ROUTINE
FEEB B020                5738            MOV     AL,EOI
FEED E620                5739            OUT     020H,AL         ; END OF INTERRUPT TO 8259
FEEF 5A                  5740            POP     DX
FEF0 58                  5741            POP     AX
FEF1 1F                  5742            POP     DS              ; RESET MACHINE STATE
FEF2 CF                  5743            IRET                    ; RETURN FROM INTERRUPT
                         5744    TIMER_INT       ENDP
                         5745    ;--------------------------------------------
                         5746    ; THESE ARE THE VECTORS WHICH ARE MOVED INTO
                         5747    ;  THE 8086 INTERUPT AREA DURING POWER ON
                         5748    ;--------------------------------------------
FEF3                     5749    VECTOR_TABLE    LABEL   WORD            ; VECTOR TABLE FOR MOVE TO INTERRUPTS
                         5750
FEF3 A5FE           R    5751            DW      OFFSET TIMER_INT        ; INTERRUPT 8
FEF5 00F0           R    5752            DW      CODE
                         5753
FEF7 87E9           R    5754            DW      OFFSET  K8_INT          ; INTERRUPT 9
FEF9 00F0           R    5755            DW      CODE
                         5756
FEFB 00000000            5757            DD      0                       ; INTERRUPT A
FEFF 00000000            5758            DD      0                       ; INTERRUPT B
FF03 00000000            5759            DD      0                       ; INTERRUPT C
FF07 00000000            5760            DD      0                       ; INTERRUPT D
                         5761
FF0B 57EF           R    5762            DW      OFFSET DISK_INT         ; INTERRUPT E
FF0D 00F0           R    5763            DW      CODE
                         5764
FF0F 00000000            5765            DD      0                       ; INTERRUPT F
                         5766
FF13 65F0           R    5767            DW      OFFSET VIDEO_IO         ; INTERRUPT 10H
FF15 00F0           R    5768            DW      CODE
                         5769
FF17 4DF8           R    5770            DW      OFFSET EQUIPMENT        ; INTERRUPT 11H
FF19 00F0           R    5771            DW      CODE
                         5772
FF1B 41F8           R    5773            DW      OFFSET MEMORY_SIZE_DETERMINE    ; INT 12H
FF1D 00F0           R    5774            DW      CODE
                         5775
FF1F 59EC           R    5776            DW      OFFSET DISKETTE_IO      ; INTERRUPT 13H
FF21 00F0           R    5777            DW      CODE
                         5778
FF23 39E7           R    5779            DW      OFFSET RS232_IO         ; INTERRUPT 14H
FF25 00F0           R    5780            DW      CODE
                         5781
FF27 59F8           R    5782            DW      OFFSET CASSETTE_IO      ; INTERRUPT 15H
FF29 00F0           R    5783            DW      CODE
                         5784
FF2B 2EE8           R    5785            DW      OFFSET KEYBOARD_IO      ; INTERRUPT 16H
FF2D 00F0           R    5786            DW      CODE
                         5787
FF2F D2EF           R    5788            DW      OFFSET PRINTER_IO       ; INTERRUPT 17H
FF31 00F0           R    5789            DW      CODE
                         5790
```



```
LOC OBJ                 LINE    SOURCE

FF33 0000                5791            DW      00000H                  ; INTERRUPT 18H
FF35 00F6                5792            DW      0F600H                  ;  ROM BASIC ENTRY POINT
                         5793
FF37 F2E6           R    5794            DW      OFFSET BOOT_STRAP       ; INTERRUPT 19H
FF39 00F0           R    5795            DW      CODE
                         5796
FF3B 6EFE           R    5797            DW      TIME_OF_DAY             ; INTERRUPT 1AH -- TIME OF DAY
FF3D 00F0           R    5798            DW      CODE
                         5799
FF3F 53FF           R    5800            DW      DUMMY_RETURN            ; INTERRUPT 1BH -- KEYBOARD BREAK ADDR
FF41 00F0           R    5801            DW      CODE
                         5802
FF43 53FF           R    5803            DW      DUMMY_RETURN            ; INTERRUPT 1C -- TIMER BREAK ADDR
FF45 00F0           R    5804            DW      CODE
                         5805
FF47 A4F0           R    5806            DW      VIDEO_PARMS             ; INTERRUPT 1D -- VIDEO PARAMETERS
FF49 00F0           R    5807            DW      CODE
                         5808
FF4B C7EF           R    5809            DW      OFFSET DISK_BASE        ; INTERRUPT 1E -- DISK PARMS
FF4D 00F0           R    5810            DW      CODE
                         5811
FF4F 00000000            5812            DD      0                       ; INTERRUPT 1F -- POINTER TO VIDEO EXT
                         5813
FF53                     5814    DUMMY_RETURN:
FF53 CF                  5815            IRET                            ; DUMMY RETURN FOR BREAK FROM KEYBOARD
                         5816    ;-- INT 5 ----------------------------------
                         5817    ;       THIS LOGIC WILL BE INVOKED BY INTERRUPT 05H TO PRINT
                         5818    ;       THE SCREEN. THE CURSOR POSITION AT THE TIME THIS ROUTINE
                         5819    ;       IS INVOKED WILL BE SAVED AND RESTORED UPON COMPLETION. THE
                         5820    ;       ROUTINE IS INTENDED TO RUN WITH INTERRUPTS ENABLED.
                         5821    ;       IF A SUBSEQUENT 'PRINT SCREEN KEY IS DEPRESSED DURING THE
                         5822    ;       TIME THIS ROUTINE IS PRINTING IT WILL BE IGNORED.
                         5823    ;       ADDRESS 50:0 CONTAINS THE STATUS OF THE PRINT SCREEN:
                         5824    ;
                         5825    ;       50:0    =0      EITHER PRINT SCREEN HAS NOT BEEN CALLED
                         5826    ;                       OR UPON RETURN FROM A CALL THIS INDICATES
                         5827    ;                       A SUCCESSFUL OPERATION.
                         5828    ;
                         5829    ;               =1      PRINT SCREEN IS IN PROGRESS
                         5830    ;
                         5831    ;               =377    ERROR ENCOUNTEREO DURING PRINTING
                         5832    ;--------------------------------------------
                         5833            ASSUME  CS:CODE,DS:XXDATA
                         5834
FF54                     5835    PRINT_SCREEN    PROC    FAR
FF54 FB                  5836            STI                     ;MUST RUN WITH INTERRUPTS ENABLED
FF55 1E                  5837            PUSH    DS              ;MUST USE 50:0 FOR DATA AREA STORAGE
FF56 50                  5838            PUSH    AX
FF57 53                  5839            PUSH    BX
FF58 51                  5840            PUSH    CX              ;WILL USE THIS LATER FOR CURSOR LIMITS
FF59 52                  5841            PUSH    DX              ;WILL HOLD CURRENT CURSOR POSITION
FF5A B85000              5842            MOV     AX,XXDATA       ;HEX 50
FF5D 8ED8                5843            MOV     DS,AX
FF5F 803E000001     R    5844            CMP     STATUS_BYTE,1   ;SEE IF PRINT ALREADY IN PROGRESS
FF64 745F                5845            JZ      EXIT            ; JUMP IF PRINT ALREADY IN PROGRESS
FF66 C606000001     R    5846            MOV     STATUS_BYTE,1   ;INDICATE PRINT NOW IN PROGRESS
FF6B B40F                5847            MOV     AH,15           ;WILL REQUEST THE CURRENT SCREEN MODE
FF6D CD10                5848            INT     10H             ;       (AL)=MODE
                         5849                                    ;       (AH)=NUMBER COLUMNS/LINE
                         5850                                    ;       (BH)=VISUAL PAGE
                         5851            ;************************************
                         5852            ;       AT THIS POINT WE KNOW THE COLUMNS/LINE ARE IN
                         5853            ;       (AX) AND THE PAGE IF APPLICABLE IS IN (BH). THE STACK
                         5854            ;       HAS DS,AX,BX,CX,DX PUSHED. (AL) HAS VIDEO MODE
                         5855            ;
                         5856            ;************************************
FF6F 8ACC                5857            MOV     CL,AH           ;WILL MAKE USE OF (CX) REGISTER TO
FF71 B519                5858            MOV     CH,25           ;CONTROL ROW & COLUMNS
FF73 E85500              5859            CALL    CRLF            ;CARRIAGE RETURN LINE FEED ROUTINE
FF76 51                  5860            PUSH    CX              ;SAVE SCREEN BOUNDS
FF77 B403                5861            MOV     AH,3            ;WILL NOW READ THE CURSOR.
FF79 CD10                5862            INT     10H             ;AND PRESERVE THE POSITION
FF7B 59                  5863            POP     CX              ;RECALL SCREEN BOUNDS
FF7C 52                  5864            PUSH    DX              ;RECALL (BH)=VISUAL PAGE
FF7D 33D2                5865            XOR     DX,DX           ;WILL SET CURSOR POSITION TO (0,0)
```

```
LOC OBJ                 LINE    SOURCE

                        5866    ;*****************************************
                        5867    ;       THE LOOP FROM PRI10 TO THE INSTRUCTION PRIOR TO PRI20
                        5868    ;       IS THE LOOP TO READ EACH CURSOR POSITION FROM THE SCREEN
                        5869    ;       AND PRINT.
                        5870    ;*****************************************
FF7F B402               5871    PRI10:  MOV     AH,2            ;TO INDICATE CURSOR SET REQUEST
FF81 CD10               5872            INT     10H             ;NEW CURSOR POSITION ESTABLISHED
FF83 B408               5873            MOV     AH,8            ;TO INDICATE READ CHARACTER
FF85 CD10               5874            INT     10H             ;CHARACTER NOW IN [AL]
FF87 0AC0               5875            OR      AL,AL           ;SEE IF VALID CHAR
FF89 7502               5876            JNZ     PRI15           ;JUMP IF VALID CHAR
FF8B B020               5877            MOV     AL,' '          ;MAKE A BLANK
FF8D                    5878    PRI15:
FF8D 52                 5879            PUSH    DX              ;SAVE CURSOR POSITION
FF8E 33D2               5880            XOR     DX,DX           ;INDICATE PRINTER 1
FF90 32E4               5881            XOR     AH,AH           ;TO INDICATE PRINT CHAR IN [AL]
FF92 CD17               5882            INT     17H             ;PRINT THE CHARACTER
FF94 5A                 5883            POP     DX              ;RECALL CURSOR POSITION
FF95 F6C425             5884            TEST    AH, 25H         ; TEST FOR PRINTER ERROR
FF98 7521               5885            JNZ     ERR10           ; JUMP IF ERROR DETECTED
FF9A FEC2               5886            INC     DL              ;ADVANCE TO NEXT COLUMN
FF9C 3ACA               5887            CMP     CL,DL           ;SEE IF AT END OF LINE
FF9E 75DF               5888            JNZ     PRI10           ;IF NOT PROCEED
FFA0 32D2               5889            XOR     DL,DL           ;BACK TO COLUMN 0
FFA2 8AE2               5890            MOV     AH,DL           ;[AH]=0
FFA4 52                 5891            PUSH    DX              ;SAVE NEW CURSOR POSITION
FFA5 E82300             5892            CALL    CRLF            ; LINE FEED CARRIAGE RETURN
FFA8 5A                 5893            POP     DX              ;RECALL CURSOR POSITION
FFA9 FEC6               5894            INC     DH              ;ADVANCE TO NEXT LINE
FFAB 3AEE               5895            CMP     CH,DH           ;FINISHED?
FFAD 75D0               5896            JNZ     PRI10           ;IF NOT CONTINUE
FFAF 5A                 5897    PRI20:  POP     DX              ;RECALL CURSOR POSITION
FFB0 B402               5898            MOV     AH,2            ;TO INDICATE CURSOR SET REQUEST
FFB2 CD10               5899            INT     10H             ;CURSOR POSITION RESTORED
FFB4 C606000000         5900            MOV     STATUS_BYTE,0   ;INDICATE FINISHED
FFB9 EB0A               5901            JMP     SHORT EXIT      ;EXIT THE ROUTINE
FFBB 5A                 5902    ERR10:  POP     DX              ;GET CURSOR POSITION
FFBC B402               5903            MOV     AH,2            ;TO REQUEST CURSOR SET
FFBE CD10               5904            INT     10H             ;CURSOR POSITION RESTORED
FFC0 C606000FF          5905    ERR20:  MOV     STATUS_BYTE,0FFH        ;INDICATE ERROR
                        5906
FFC5 5A                 5907    EXIT:   POP     DX              ;RESTORE ALL THE REGISTERS USED
FFC6 59                 5908            POP     CX
FFC7 5B                 5909            POP     BX
FFC8 58                 5910            POP     AX
FFC9 1F                 5911            POP     DS
FFCA CF                 5912            IRET
                        5913    PRINT_SCREEN    ENDP
                        5914
                        5915    ;------ CARRIAGE RETURN, LINE FEED SUBROUTINE
                        5916
FFCB                    5917    CRLF    PROC    NEAR
FFCB 33D2               5918            XOR     DX,DX           ;PRINTER 0
FFCD 32E4               5919            XOR     AH,AH           ;WILL NOW SEND INITIAL LF,CR TO PRINTER
FFCF B00A               5920            MOV     AL,12Q          ;LF
FFD1 CD17               5921            INT     17H             ;SEND THE LINE FEED
FFD3 32E4               5922            XOR     AH,AH           ;NOW FOR THE CR
FFD5 B00D               5923            MOV     AL,15Q          ;CR
FFD7 CD17               5924            INT     17H             ;SEND THE CARRIAGE RETURN
FFD9 C3                 5925            RET
                        5926    CRLF    ENDP
                        5927    CODE    ENDS
                        5928
                        5929    ;--------------------------------
                        5930    ; POWER ON RESET VECTOR
                        5931    ;--------------------------------
FFFF                    5932    VECTOR  SEGMENT AT 0FFFFH
                        5933
                        5934    ;------ POWER ON RESET
                        5935
0000 EA5B0000F0     R   5936            JMP     RESET
                        5937
0005 30342F32342F38     5938            DB      '04/24/81'              ; RELEASE MARKER
     31
                        5939    VECTOR  ENDS
                        5940            END
```

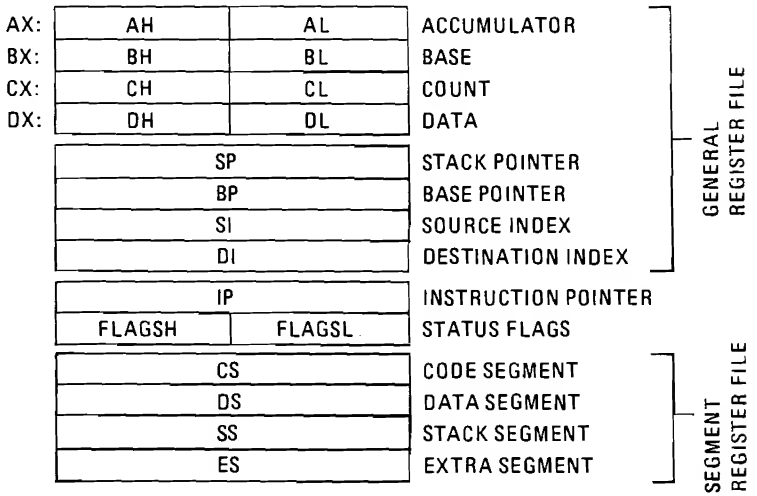

Notes for the BIOS Listing

1. The wait loop for the printer times out on form feed of > 51 lines. – line ref (3069)
2. Mode controls for the 320 x 200 video have Color/BW reversed. – line ref (3338)
3. The RS232 Timeout is 80 decimal, not 80 hexadecimal. – line ref (1566)
4. The Base Pointer register is destroyed by some video calls.
5. D_04 ◊ character in the character generator has 08 as its last value, S/80. – line ref (5511)
6. If you hit print screen in the Color/Graphics 80x25 Character Mode, the screen may not display during the print cycle.

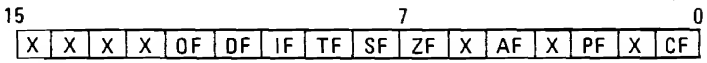
NOTES

APPENDIX B. ASSEMBLY INSTRUCTION SET REFERENCE

8088
REGISTER MODEL



Instructions which reference the flag register file as a 16-bit object use the symbol FLAGS to represent the file:



X = Don't Care

AF: AUXILIARY CARRY – BCD
CF: CARRY FLAG
PF: PARITY FLAG
SF: SIGN FLAG
ZF: ZERO FLAG

— 8080 FLAGS

DF: DIRECTION FLAG (STRINGS)
IF: INTERRUPT ENABLE FLAG
OF: OVERFLOW FLAG (CF ⊕ SF)
TF: TRAP – SINGLE STEP FLAG

— 8088 FLAGS

OPERAND SUMMARY

"reg" field Bit Assignments:

16-Bit (w=1)	8-Bit (w=0)	Segment
000 AX	000 AL	00 ES
001 CX	001 CL	01 CS
010 DX	010 DL	10 SS
011 BX	011 BL	11 DS
100 SP	100 AH	
101 BP	101 CH	
110 SI	110 DH	
111 DI	111 BH	

SECOND INSTRUCTION BYTE SUMMARY

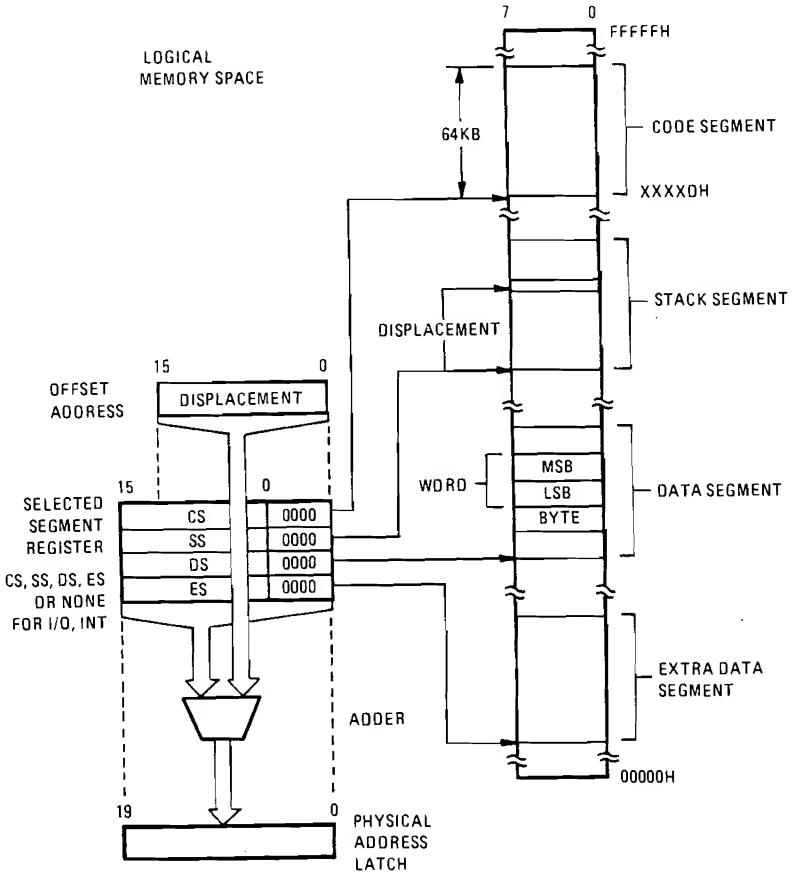
mod	xxx	r/m

mod	Displacement
00	DISP = 0*, disp-low and disp-high are absent
01	DISP = disp-low sign-extended to 16-bits, disp-high is absent
10	DISP = disp-high: disp-low
11	r/m is treated as a "reg" field

r/m	Operand Address
000	(BX) + (SI) + DISP
001	(BX) + (DI) + DISP
010	(BP) + (SI) + DISP
011	(BP) + (DI) + DISP
100	(SI) + DISP
101	(DI) + DISP
110	(BP) + DISP*
111	(BX) + DISP

DISP follows 2nd byte of instruction (before data if required).
*except if mod = 00 and r/m = 110 then EA = disp-high: disp-low.

MEMORY SEGMENTATION MODEL



SEGMENT OVERRIDE PREFIX

0	0	1	reg	1	1	0

USE OF SEGMENT OVERRIDE

OPERAND REGISTER	DEFAULT	WITH OVERRIDE PREFIX
IP (code address)	CS	Never
SP (stack address)	SS	Never
BP (stack address or stack marker)	SS	BP + DS or ES, or CS
SI or DI (not incl. strings)	DS	ES, SS, or CS
SI (implicit source addr for strings)	DS	ES, SS, or CS
DI (implicit dest addr for strings)	ES	Never

DATA TRANSFER

MOV = Move

Register/memory to/ from register

1	0	0	0	1	0	d	w	mod	reg	r/m

Immediate to register/memory

1	1	0	0	0	1	1	w	mod	0	0	0	r/m	data	data if w=1

Immediate to register

1	0	1	1	w	reg	data	data if w=1

Memory to accumulator

1	0	1	0	0	0	0	w	addr-low	addr-high

Accumulator to memory

1	0	1	0	0	0	1	w	addr-low	addr-high

Register/memory to segment register

1	0	0	0	1	1	1	0	mod	0	reg	r/m

Segment register to register/memory

1	0	0	0	1	1	0	0	mod	0	reg	r/m

PUSH = Push

Register/memory

1	1	1	1	1	1	1	1	mod	1	1	0	r/m

Register

0	1	0	1	0	reg

Segment register

0	0	0	reg	1	1	0

POP = Pop

Register/memory

1	0	0	0	1	1	1	1	mod	0	0	0	r/m

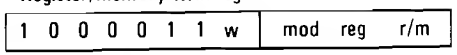
Register

0	1	0	1	1	reg

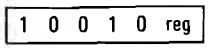
Segment register

0	0	0	reg	1	1	1

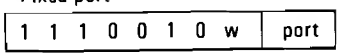
XCHG = Exchange
Register/memory with register



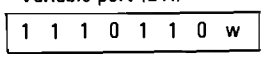
Register with accumulator



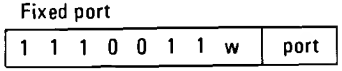
IN = Input to AL/AX from
Fixed port



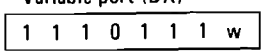
Variable port (DX)



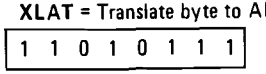
OUT = Output from AL/AX to
Fixed port



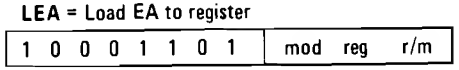
Variable port (DX)



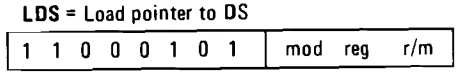
XLAT = Translate byte to AL



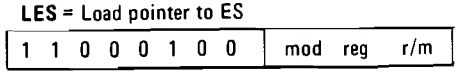
LEA = Load EA to register



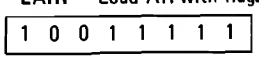
LDS = Load pointer to DS



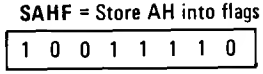
LES = Load pointer to ES



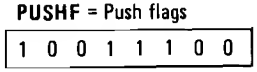
LAHF = Load AH with flags



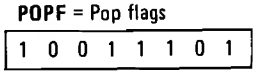
SAHF = Store AH into flags



PUSHF = Push flags

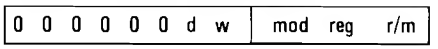


POPF = Pop flags

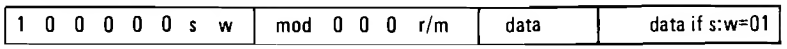


ARITHMETIC

ADD = Add
Reg./memory with register to either

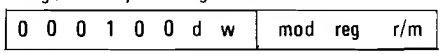


Immediate to register/memory

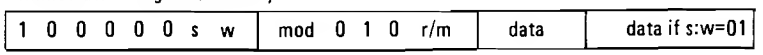


Immediate to accumulator

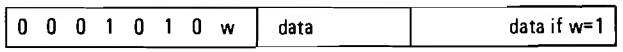
ADC = Add with carry
Reg./memory with register to either



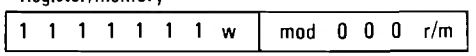
Immediate to register/memory



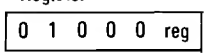
Immediate to accumulator



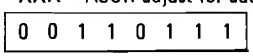
INC = Increment
Register/memory



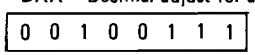
Register



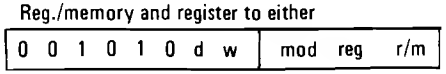
AAA = ASCII adjust for add



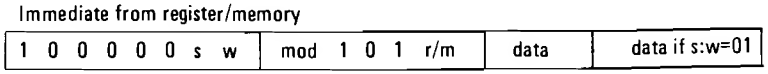
DAA = Decimal adjust for add



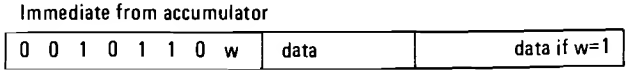
SUB = Subtract
Reg./memory and register to either



Immediate from register/memory



Immediate from accumulator



SBB = Subtract with borrow
Reg./memory and register to either

0 0 0 1 1 0 d w	mod reg r/m

Immediate from register/memory

1 0 0 0 0 0 s w	mod 0 1 1 r/m	data	data if s:w=01

Immediate from accumulator

0 0 0 1 1 1 0 w	data	data if w=1

DEC = Decrement
Register/memory

1 1 1 1 1 1 1 w	mod 0 0 1 r/m

Register

0 1 0 0 1 reg

NEG = Change sign

1 1 1 1 0 1 1 w	mod 0 1 1 r/m

CMP = Compare
Register/memory and register

0 0 1 1 1 0 d w	mod reg r/m

Immediate with register/memory

1 0 0 0 0 0 s w	mod 1 1 1 r/m	data	data if s:w=01

Immediate with accumulator

0 0 1 1 1 1 0 w	data	data if w=1

AAS = ASCII adjust for subtract

0 0 1 1 1 1 1 1

DAS = Decimal adjust for subtract

0 0 1 0 1 1 1 1

MUL = Multiply (unsigned)

1 1 1 1 0 1 1 w	mod 1 0 0 r/m

IMUL = Integer multiply (signed)

1 1 1 1 0 1 1 w	mod 1 0 1 r/m

AAM = ASCII adjust for multiply

1 1 0 1 0 1 0 0	0 0 0 0 1 0 1 0

DIV = Divide (unsigned)

1 1 1 1 0 1 1 w	mod 1 1 0 r/m

IDIV = Integer divide (signed)

1 1 1 1 0 1 1 w	mod 1 1 1 r/m

AAD = ASCII adjust for divide

1 1 0 1 0 1 0 1	0 0 0 0 1 0 1 0

CBW = Convert byte to word

1 0 0 1 1 0 0 0

CWD = Convert word to double word

1 0 0 1 1 0 0 1

LOGIC

NOT = Invert

1 1 1 1 0 1 1 w	mod 0 1 0 r/m

SHL/SAL = Shift logical/arithmetic left

1 1 0 1 0 0 v w	mod 1 0 0 r/m

SHR = Shift logical right

1 1 0 1 0 0 v w	mod 1 0 1 r/m

SAR = Shift arithmetic right

1 1 0 1 0 0 v w	mod 1 1 1 r/m

ROL = Rotate left

1 1 0 1 0 0 v w	mod 0 0 0 r/m

ROR = Rotate right

1 1 0 1 0 0 v w	mod 0 0 1 r/m

RCL = Rotate through carry left

1 1 0 1 0 0 v w	mod 0 1 0 r/m

RCR = Rotate through carry right

1 1 0 1 0 0 v w	mod 0 1 1 r/m

AND = And
Reg./memory and register to either

0 0 1 0 0 0 d w	mod reg r/m

Immediate to register/memory

1 0 0 0 0 0 0 w	mod 1 0 0 r/m	data	data if w=1

Immediate to accumulator

0 0 1 0 0 1 0 w	data	data if w=1

TEST = And function to flags, no result
Register/memory and register

1 0 0 0 0 1 0 w	mod reg r/m

Immediate data and register/memory

1 1 1 1 0 1 1 w	mod 0 0 0 r/m	data	data if w=1

Immediate data and accumulator

1 0 1 0 1 0 0 w	data	data if w=1

OR = Or
Reg./memory and register to either

0 0 0 0 1 0 d w	mod reg r/m

Immediate to register/memory

1 0 0 0 0 0 0 w	mod 0 0 1 r/m	data	data if w=1

Immediate to accumulator

0 0 0 0 1 1 0 w	data	data if w=1

XOR = Exclusive or
Reg./memory and register to either

0 0 1 1 0 0 d w	mod reg r/m

Immediate to register/memory

1 0 0 0 0 0 0 w	mod 1 1 0 r/m	data	data if w=1

Immediate to accumulator

0 0 1 1 0 1 0 w	data	data if w=1

STRING MANIPULATION

REP = Repeat

1 1 1 1 0 0 1 z

MOVS = Move String

1 0 1 0 0 1 0 w

CMPS = Compare String

1 0 1 0 0 1 1 w

SCAS = Scan String

1 0 1 0 1 1 1 w

LODS = Load String

1 0 1 0 1 1 0 w

STOS = Store String

1 0 1 0 1 0 1 w

CONTROL TRANSFER

CALL = Call
Direct within segment

1 1 1 0 1 0 0 0	disp-low	disp-high

Indirect within segment

1 1 1 1 1 1 1 1	mod 0 1 0 r/m

Direct intersegment

1 0 0 1 1 0 1 0	offset-low	offset-high
	seg-low	seg-high

Indirect intersegment

1 1 1 1 1 1 1 1	mod 0 1 1 r/m

JMP = Unconditional Jump
Direct within segment

1 1 1 0 1 0 0 1	disp-low	disp-high

Direct within segment-short

1 1 1 0 1 0 1 1	disp

Indirect within segment

1 1 1 1 1 1 1 1	mod 1 0 0 r/m

Direct intersegment

1 1 1 0 1 0 1 0	offset-low	offset-high
	seg-low	seg-high

Indirect intersegment

1 1 1 1 1 1 1 1	mod 1 0 1 r/m

RET = Return from CALL
Within segment

1 1 0 0 0 0 1 1

Within seg. adding immed to SP

1 1 0 0 0 0 1 0	data-low	data-high

Intersegment

1 1 0 0 1 0 1 1

Intersegment, adding immediate to SP

1 1 0 0 1 0 1 0	data-low	data-high

JE/JZ = Jump on equal/zero

0 1 1 1 0 1 0 0	disp

JL/JNGE = Jump on less/not greater or equal

0 1 1 1 1 1 0 0	disp

JLE/JNG = Jump on less or equal/not greater

0 1 1 1 1 1 1 0	disp

JB/JNAE = Jump on below/not above or equal

0 1 1 1 0 0 1 0	disp

JBE/JNA = Jump on below or equal/not above

0 1 1 1 0 1 1 0	disp

JP/JPE = Jump on parity/parity even

0 1 1 1 1 0 1 0	disp

JO = Jump on overflow

0 1 1 1 0 0 0 0	disp

JS = Jump on sign

0 1 1 1 1 0 0 0	disp

JNE/JNZ = Jump on not equal/not zero

0 1 1 1 0 1 0 1	disp

JNL/JGE = Jump on not less/greater or equal

0 1 1 1 1 1 0 1	disp

JNLE/JG = Jump on not less or equal/greater

0 1 1 1 1 1 1 1	disp

JNB/JAE = Jump on not below/above or equal

0 1 1 1 0 0 1 1	disp

JNBE/JA = Jump on not below or equal/above

0 1 1 1 0 1 1 1	disp

JNP/JPO = Jump on not parity/parity odd

0 1 1 1 1 0 1 1	disp

JNO = Jump on not overflow

0 1 1 1 0 0 0 1	disp

JNS = Jump on not sign

0 1 1 1 1 0 0 1	disp

LOOP = Loop CX times

1 1 1 0 0 0 1 0	disp

LOOPZ/LOOPE = Loop while zero/equal

1 1 1 0 0 0 0 1	disp

LOOPNZ/LOOPNE = Loop while not zero/not equal

1 1 1 0 0 0 0 0	disp

JCXZ = Jump on CX zero

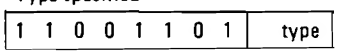
1 1 1 0 0 0 1 1	disp

8088 CONDITIONAL TRANSFER OPERATIONS

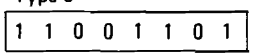
Instruction	Condition	Interpretation
JE or JZ	ZF = 1	"equal" or "zero"
JL or JNGE	(SF xor OF) = 1	"less" or "not greater or equal"
JLE or JNG	((SP xor OF) or ZF) = 1	"less or equal" or "not greater"
JB or JNAE	CF = 1	"below" or "not above or equal"
JBE or JNA	(CF or ZF) = 1	"below or equal" or "not above"
JP or JPE	PF = 1	"parity" or "parity even"
JO	OF = 1	"overflow"
JS	SF = 1	"sign"
JNE or JNZ	ZF = 0	"not equal" or "not zero"
JNL or JGE	(SF xor OF) = 0	"not less" or "greater or equal"
JNLE or JG	((SF xor OF) or ZF) = 0	"not less or equal" or "greater"
JNB or JAE	CF = 0	"not below" or "above or equal"
JNBE or JA	(CF or ZF) = 0	"not below or equal" or "above"
JNP or JPO	PF = 0	"not parity" or "parity odd"
JNO	OF = 0	"not overflow"
JNS	SF = 0	"not sign"

*"Above" and "below" refer to the relation between two unsigned values, while "greater" and "less" refer to the relation between two signed values.

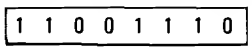
INT = Interrupt
Type specified



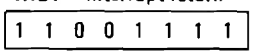
Type 3



INTO = Interrupt on overflow

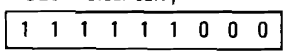


IRET = Interrupt return

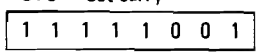


PROCESSOR CONTROL

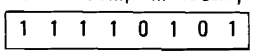
CLC = Clear carry



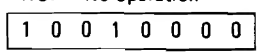
STC = Set carry



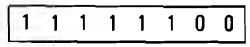
CMC = Complement carry



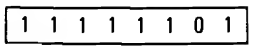
NOP = No operation



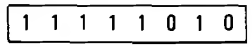
CLD = Clear direction



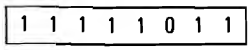
STD = Set direction



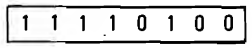
CLI = Clear interrupt



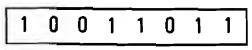
STI = Set interrupt



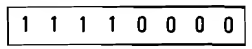
HLT = Halt



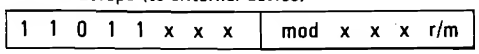
WAIT = Wait



LOCK = Bus lock prefix



ESC = Escape (to external device)



Footnotes:
if d = 1 then "to"; if d = 0 then "from"
if w = 1 then word instruction; if w = 0 then byte instruction
if s:w = 01 then 16 bits of immediate data from the operand
if s:w = 11 then an immediate data byte is sign extended to form the 16-bit operand
if v = 0 then "count" = 1; if v = 1 then "count" in (CL)
x = don't care
z is used for some string primitives to compare with ZF FLAG
AL = 8-bit accumulator
AX = 16-bit accumulator
CX = Count register
DS = Data segment
DX = Variable port register
ES = Extra segment
Above/below refers to unsigned value
Greater = more positive;
Less = less positive (more negative) signed values

8088 INSTRUCTION SET MATRIX

HI \ LO	0	1	2	3	4	5	6	7
0	ADD b.f.r/m	ADD w.f.r/m	ADD b,t,r/m	ADD w,t,r/m	ADD b.ia	Add w.ia	PUSH ES	POP ES
1	ADC b.f.r/m	ADC w.f.r/m	ADC b,t,r/m	ADC w,t,r/m	ADC b,i	ADC w,i	PUSH SS	POP SS
2	AND b.f.r/m	AND w.f.r/m	AND b,t,r/m	AND w,t,r/m	AND b,i	AND w.i	SEG =ES	DAA
3	XOR b.f.r/m	XOR w,f,r/m	XOR b,t,r/m	XOR w,t,r/m	XOR b,i	XOR w.i	SEG =SS	AAA
4	INC AX	INC CX	INC OX	INC BX	INC SP	INC BP	INC SI	INC OI
5	PUSH AX	PUSH CX	PUSH OX	PUSH BX	PUSH SP	PUSH BP	PUSH SI	PUSH OI
6								
7	JO	JNO	JB/ JNAE	JNB/ JAE	JE/ JZ	JNE/ JNZ	JBE/ JNA	JNBE/ JA
8	Immed b.r/m	Immed w.r/m	Immed b.r/m	Immed is.r/m	TEST b.r/m	TEST w.r/m	XCHG b.r/m	XCHG w.r/m
9	NOP	XCHG CX	XCHG OX	XCHG BX	XCHG SP	XCHG BP	XCHG SI	XCHG OI
A	MOV m→AL	MOV m→AX	MOV AL→m	MOV AX→m	MOVS b	MOVS w	CMPS b	CMPS w
B	MOV i→AL	MOV i→CL	MOV i→DL	MOV i→BL	MOV i→AH	MOV i→CH	MOV i→DH	MOV i→BH
C			RET. (i+SP)	RET	LES	LDS	MOV b.i.r/m	MOV w.i.r/m
D	Shift b	Shift w	Shift b.v	Shift w.v	AAM	AAD		XLAT
E	LOOPNZ/ LOOPNE	LOOPZ/ LOOPE	LOOP	JCXZ	IN b	IN w	OUT b	OUT w
F	LOCK		REP	REP Z	HLT	CMC	Grp 1 b.r/m	Grp 1 w.r/m

b = byte operation
d = direct
f = from CPU reg
i = immediate
ia = immed. to accum.
id = indirect
is = immed. byte, sign ext.
l = long ie. intersegment

m = memory
r/m = EA is second byte
si = short intrasegment
sr = segment register
t = to CPU reg
v = variable
w = word operation
z = zero

8088 INSTRUCTION SET MATRIX

HI \ LO	8	9	A	B	C	D	E	F
0	OR b.f.r/m	OR w.f.r/m	OR b,t,r/m	OR w,t,r/m	OR b.i	OR w.i	PUSH CS	
1	SBB b.f.r/m	SBB w.f.r/m	SBB b,t,r/m	SBB w,t,r/m	SBB b.i	SBB w.i	PUSH DS	POP DS
2	SUB b.f.r/m	SUB w.f.r/m	SUB b,t,r/m	SUB w,t,r/m	SUB b.i	SUB w.i	SEG CS	DAS
3	CMP b.f.r/m	CMP w.f.r/m	CMP b,t,r/m	CMP w,t,r/m	CMP b.i	CMP w.i	SEG DS	AAS
4	DEC AX	DEC CX	DEC OX	DEC BX	DEC SP	DEC BP	DEC SI	DEC DI
5	POP AX	POP CX	POP OX	POP BX	POP SP	POP BP	POP SI	POP DI
6								
7	JS	JNS	JP/ JPE	JNP/ JPO	JL/ JNGE	JNL/ JGE	JLE/ JNG	JNLE/ JG
8	MOV b.f.r/m	MOV w.f.r/m	MOV b,t,r/m	MOV w,t,r/m	MOV sr.t.r/m	LEA	MOV sr.f.r/m	POP r/m
9	CBW	CWD	CALL l.d	WAIT	PUSHF	POPF	SAHF	LAHF
A	TEST b.i	TEST w.i	STOS b	STOS w	LODS b	LODS w	SCAS b	SCAS w
B	MOV i→AX	MOV i→CX	MOV i→OX	MOV i→BX	MOV i→SP	MOV i→BP	MOV i→SI	MOV i→DI
C			RET l,(i+SP)	RET l	INT Type 3	INT (Any)	INTO	IRET
D	ESC 0	ESC 1	ESC 2	ESC 3	ESC 4	ESC 5	ESC 6	ESC 7
E	CALL d	JMP d	JMP l.d	JMP si.d	IN v.b	IN v.w	OUT v.b	OUT v.w
F	CLC	STC	CLI	STI	CLD	STD	Grp 2 b.r/m	Grp 2 w.r/m

where

mod ☐ r/m	000	001	010	011	100	101	100	111
Immed	ADD	OR	ADC	SBB	AND	SUB	XOR	CMP
Shift	ROL	ROR	RCL	RCR	SHL/SAL	SHR		SAR
Grp 1	TEST		NOT	NEG	MUL	IMUL	DIV	IDIV
Grp 2	INC	DEC	CALL id	CALL l,id	JMP id	JMP l,id	PUSH	

INSTRUCTION SET INDEX

Appendix C. Of Characters Keystrokes and Color (U.S. Keyboard Layout

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
00	0	Blank (Null)	Ctrl 2		Black	Black	Non-Display
01	1	☺	Ctrl A		Black	Blue	Underline
02	2	☻	Ctrl B		Black	Green	Normal
03	3	♥	Ctrl C		Black	Cyan	Normal
04	4	♦	Ctrl D		Black	Red	Normal
05	5	♣	Ctrl E		Black	Magenta	Normal
06	6	♠	Ctrl F		Black	Brown	Normal
07	7	•	Ctrl G		Black	Light Grey	Normal
08	8	■	Ctrl H, Backspace, Shift Backspace		Black	Dark Grey	Non-Display
09	9	○	Ctrl I		Black	Light Blue	High Intensity Underline
0A	10	○	Ctrl J, Ctrl↵		Black	Light Green	High Intensity
0B	11	♂	Ctrl K		Black	Light Green	High Intensity
0C	12	♀	Ctrl L,		Black	Light Red	High Intensity
0D	13	♪	Ctrl M, ↵, Shift↵		Black	Light Magenta	High Intensity
0E	14	♫	Ctrl N		Black	Yellow	High Intensity
0F	15	☼	Ctrl O		Black	White	High Intensity
10	16	►	Ctrl P		Blue	Black	Normal
11	17	◄	Ctrl Q		Blue	Blue	Underline
12	18	↕	Ctrl R		Blue	Green	Normal
13	19	‼	Ctrl S		Blue	Cyan	Normal
14	20	¶	Ctrl T		Blue	Red	Normal
15	21	§	Ctrl U			Magenta	Normal
16	22	▬	Ctrl V		Blue	Brown	Normal
17	23	↨	Ctrl W		Blue	Light Grey	Normal

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
18	24	↑	Ctrl X		Blue	Dark Grey	High Intensity
19	25	↓	Ctrl Y		Blue	Light Blue	High Intensity Underline
1A	26	→	Ctrl Z		Blue	Light Green	High Intensity
1B	27	←	Ctrl [, Esc, Shift Esc, Ctrl Esc		Blue	Light Cyan	High Intensity
1C	28	└	Ctrl \		Blue	Light Red	High Intensity
1D	29	↔	Ctrl]		Blue	Light Magenta	High Intensity
1E	30	▲	Ctrl 6		Blue	Yellow	High Intensity
1F	31	▼	Ctrl –		Blue	White	High Intensity
20	32	Blank Space	Space Bar, Shift, Space, Ctrl Space, Alt Space		Green	Black	Normal
21	33	!	!	Shift	Green	Blue	Underline
22	34	"	"	Shift	Green	Green	Normal
23	35	#	#	Shift	Green	Cyan	Normal
24	36	$	$	Shift	Green	Red	Normal
25	37	%	%	Shift	Green	Magenta	Normal
26	38	&	&	Shift	Green	Brown	Normal
27	39	'	'		Green	Light Grey	Normal
28	40	(	(	Shift	Green	Dark Grey	High Intensity
29	41	)	)	Shift	Green	Light Blue	High Intensity Underline
2A	42	*	*	Note 1	Green	Light Green	High Intensity
2B	43	+	+	Shift	Green	Light Cyan	High Intensity
2C	44	,	,		Green	Light Red	High Intensity
2D	45	—	—		Green	Light Magenta	High Intensity
2E	46	.	.	Note 2	Green	Yellow	High Intensity

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
2F	47	/	/		Green	White	High Intensity
30	48	0	0	Note 3	Cyan	Black	Normal
31	49	1	1	Note 3	Cyan	Blue	Underline
32	50	2	2	Note 3	Cyan	Green	Normal
33	51	3	3	Note 3	Cyan	Cyan	Normal
34	52	4	4	Note 3	Cyan	Red	Normal
35	53	5	5	Note 3	Cyan	Magenta	Normal
36	54	6	6	Note 3	Cyan	Brown	Normal
37	55	7	7	Note 3	Cyan	Light Grey	Normal
38	56	8	8	Note 3	Cyan	Dark Grey	High Intensity
39	57	9	9	Note 3	Cyan	Light Blue	High Intensity Underline
3A	58	:	:	Shift	Cyan	Light Green	High Intensity
3B	59	;	;		Cyan	Light Cyan	High Intensity
3C	60	<	<	Shift	Cyan	Light Red	High Intensity
3D	61	=	=		Cyan	Light Magenta	High Intensity
3E	62	>	>	Shift	Cyan	Yellow	High Intensity
3F	63	?	?	Shift	Cyan	White	High Intensity
40	64	@	@	Shift	Red	Black	Normal
41	65	A	A	Note 4	Red	Blue	Underline
42	66	B	B	Note 4	Red	Green	Normal
43	67	C	C	Note 4	Red	Cyan	Normal
44	68	D	D	Note 4	Red	Red	Normal
45	69	E	E	Note 4	Red	Magenta	Normal
46	70	F	F	Note 4	Red	Brown	Normal
47	71	G	G	Note 4	Red	Light Grey	Normal
48	72	H	H	Note 4	Red	Dark Grey	High Intensity
49	73	I	I	Note 4	Red	Light Blue	High Intensity Underline
4A	74	J	J	Note 4	Red	Light Green	High Intensity

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
4B	75	K	K	Note 4	Red	Light Cyan	High Intensity
4C	76	L	L	Note 4	Red	Light Red	High Intensity
4D	77	M	M	Note 4	Red	Light Magenta	High Intensity
4E	78	N	N	Note 4	Red	Yellow	High Intensity
4F	79	O	O	Note 4	Red	White	High Intensity
50	80	P	P	Note 4	Magenta	Black	Normal
51	81	Q	Q	Note 4	Magenta	Blue	Underline
52	82	R	R	Note 4	Magenta	Green	Normal
53	83	S	S	Note 4	Magenta	Cyan	Normal
54	84	T	T	Note 4	Magenta	Red	Normal
55	85	U	U	Note 4	Magenta	Magenta	Normal
56	86	V	V	Note 4	Magenta	Brown	Normal
57	87	W	W	Note 4	Magenta	Light Grey	Normal
58	88	X	X	Note 4	Magenta	Dark Grey	High Intensity
59	89	Y	Y	Note 4	Magenta	Light Blue	High Intensity Underline
5A	90	Z	Z	Note 4	Magenta	Light Green	High Intensity
5B	91	[	[		Magenta	Light Cyan	High Intensity
5C	92	\	\		Magenta	Light Red	High Intensity
5D	93	]	]		Magenta	Light Magenta	High Intensity
5E	94	^	^	Shift	Magenta	Yellow	High Intensity
5F	95	—	—	Shift	Magenta	White	High Intensity
60	96	`	`		Yellow	Black	Normal
61	97	a	a	Note 5	Yellow	Blue	Underline
62	98	b	b	Note 5	Yellow	Green	Normal
63	99	c	c	Note 5	Yellow	Cyan	Normal
64	100	d	d	Note 5	Yellow	Red	Normal
65	101	e	e	Note 5	Yellow	Magenta	Normal
66	102	f	f	Note 5	Yellow	Brown	Normal

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
67	103	g	g	Note 5	Yellow	Light Grey	Normal
68	104	h	h	Note 5	Yellow	Dark Grey	High Intensity
69	105	i	i	Note 5	Yellow	Light Blue	High Intensity Underline
6A	106	j	j	Note 5	Yellow	Light Green	High Intensity
6B	107	k	k	Note 5	Yellow	Light Cyan	High Intensity
6C	108	l	l	Note 5	Yellow	Light Red	High Intensity
6D	109	m	m	Note 5	Yellow	Light Magenta	High Intensity
6E	110	n	n	Note 5	Yellow	Yellow	High Intensity
6F	111	o	o	Note 5	Yellow	White	High Intensity
70	112	p	p	Note 5	White	Black	Reverse Video
71	113	q	q	Note 5	White	Blue	Underline
72	114	r	r	Note 5	White	Green	Normal
73	115	s	s	Note 5	White	Cyan	Normal
74	116	f	f	Note 5	White	Red	Normal
75	117	u	u	Note 5	White	Magenta	Normal
76	118	v	v	Note 5	White	Brown	Normal
77	119	w	w	Note 5	White	Light Grey	Normal
78	120	x	x	Note 5	White	Dark Grey	Reverse Video
79	121	y	y	Note 5	White	Light Blue	High Intensity Underline
7A	122	z	z	Note 5	White	Light Green	High Intensity
7B	123	{	{	Shift	White	Light Cyan	High Intensity
7C	124	¦	¦	Shift	White	Light Red	High Intensity
7D	125	}	}	Shift	White	Light Magenta	High Intensity
7E	126	~	~	Shift	White	Yellow	High Intensity
7F	127	Δ	Ctrl —		White	White	High Intensity

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
* * * *		80 to FF Hex are Flashing in both Color & IBM Monochrome * * * *					
80	128	Ç	Alt 128	Note 6	Black	Black	Non-Display
81	129	ü	Alt 129	Note 6	Black	Blue	Underline
82	130	é	Alt 130	Note 6	Black	Green	Normal
83	131	â	Alt 131	Note 6	Black	Cyan	Normal
84	132	ä	Alt 132	Note 6	Black	Red	Normal
85	133	à	Alt 133	Note 6	Black	Magenta	Normal
86	134	å	Alt 134	Note 6	Black	Brown	Normal
87	135	ç	Alt 135	Note 6	Black	Light Grey	Normal
88	136	ê	Alt 136	Note 6	Black	Dark Grey	Non-Display
89	137	ë	Alt 137	Note 6	Black	Light Blue	High Intensity Underline
8A	138	è	Alt 138	Note 6	Black	Light Green	High Intensity
8B	139	ï	Alt 139	Note 6	Black	Light Cyan	High Intensity
8C	140	î	Alt 140	Note 6	Black	Light Red	High Intensity
8D	141	ì	Alt 141	Note 6	Black	Light Magenta	High Intensity
8E	142	Ä	Alt 142	Note 6	Black	Yellow	High Intensity
8F	143	Å	Alt 143	Note 6	Black	White	High Intensity
90	144	É	Alt 144	Note 6	Blue	Black	Normal
91	145	æ	Alt 145	Note 6	Blue	Blue	Underline
92	146	Æ	Alt 146	Note 6	Blue	Green	Normal
93	147	ô	Alt 147	Note 6	Blue	Cyan	Normal
94	148	ö	Alt 148	Note 6	Blue	Red	Normal
95	149	ò	Alt 149	Note 6	Blue	Magenta	Normal
96	150	û	Alt 150	Note 6	Blue	Brown	Normal
97	151	ù	Alt 151	Note 6	Blue	Light Grey	Normal
98	152	ÿ	Alt 152	Note 6	Blue	Dark Grey	High Intensity
99	153	Ö	Alt 153	Note 6	Blue	Light Blue	High Intensity Underline
9A	154	Ü	Alt 154	Note 6	Blue	Light Green	High Intensity

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
9B	155	¢	Alt 155	Note 6	Blue	Light Cyan	High Intensity
9C	156	£	Alt 156	Note 6	Blue	Light Red	High Intensity
9D	157	¥	Alt 157	Note 6	Blue	Light Magenta	High Intensity
9E	158	Pt	Alt 158	Note 6	Blue	Yellow	High Intensity
9F	159	ƒ	Alt 159	Note 6	Blue	White	High Intensity
A0	160	á	Alt 160	Note 6	Green	Black	Normal
A1	161	í	Alt 161	Note 6	Green	Blue	Underline
A2	162	ó	Alt 162	Note 6	Green	Green	Normal
A3	163	ú	Alt 163	Note 6	Green	Cyan	Normal
A4	164	ñ	Alt 164	Note 6	Green	Red	Normal
A5	165	Ñ	Alt 165	Note 6	Green	Magenta	Normal
A6	166	a	Alt 166	Note 6	Green	Brown	Normal
A7	167	o	Alt 167	Note 6	Green	Light Grey	Normal
A8	168	¿	Alt 168	Note 6	Green	Dark Grey	High Intensity
A9	169	⌐	Alt 169	Note 6	Green	Light Blue	High Intensity Underline
AA	170	¬	Alt 170	Note 6	Green	Light Green	High Intensity
AB	171	½	Alt 171	Note 6	Green	Light Cyan	High Intensity
AC	172	¼	Alt 172	Note 6	Green	Light Red	High Intensity
AD	173	¡	Alt 173	Note 6	Green	Light Magenta	High Intensity
AE	174	<<	Alt 174	Note 6	Green	Yellow	High Intensity
AF	175	>>	Alt 175	Note 6	Green	White	High Intensity
B0	176	▒	Alt 176	Note 6	Cyan	Black	Normal
B1	177	▒	Alt 177	Note 6	Cyan	Blue	Underline
B2	178	▓	Alt 178	Note 6	Cyan	Green	Normal
B3	179	│	Alt 179	Note 6	Cyan	Cyan	Normal
B4	180	┤	Alt 180	Note 6	Cyan	Red	Normal
B5	181	╡	Alt 181	Note 6	Cyan	Magenta	Normal
B6	182	╢	Alt 182	Note 6	Cyan	Brown	Normal

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
B7	183		Alt 183	Note 6	Cyan	Light Grey	Normal
B8	184		Alt 184	Note 6	Cyan	Dark Grey	High Intensity
B9	185		Alt 185	Note 6	Cyan	Light Blue	High Intensity Underline
BA	186		Alt 186	Note 6	Cyan	Light Green	High Intensity
BB	187		Alt 187	Note 6	Cyan	Light Cyan	High Intensity
BC	188		Alt 188	Note 6	Cyan	Light Red	High Intensity
BD	189		Alt 189	Note 6	Cyan	Light Magenta	High Intensity
BE	190		Alt 190	Note 6	Cyan	Yellow	High Intensity
BF	191		Alt 191	Note 6	Cyan	White	High Intensity
C0	192		Alt 192	Note 6	Red	Black	Normal
C1	193		Alt 193	Note 6	Red	Blue	Underline
C2	194		Alt 194	Note 6	Red	Green	Normal
C3	195		Alt 195	Note 6	Red	Cyan	Normal
C4	196		Alt 196	Note 6	Red	Red	Normal
C5	197		Alt 197	Note 6	Red	Magenta	Normal
C6	198		Alt 198	Note 6	Red	Brown	Normal
C7	199		Alt 199	Note 6	Red	Light Grey	Normal
C8	200		Alt 200	Note 6	Red	Dark Grey	High Intensity
C9	201		Alt 201	Note 6	Red	Light Blue	High Intensity Underline
CA	202		Alt 202	Note 6	Red	Light Green	High Intensity
CB	203		Alt 203	Note 6	Red	Light Cyan	High Intensity
CC	204		Alt 204	Note 6	Red	Light Red	High Intensity
CD	205		Alt 205	Note 6	Red	Light Magenta	High Intensity
CE	206		Alt 206	Note 6	Red	Yellow	High Intensity
CF	207		Alt 207	Note 6	Red	White	High Intensity
D0	208		Alt 208	Note 6	Magenta	Black	Normal

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
D1	209		Alt 209	Note 6	Magenta	Blue	Underline
D2	210		Alt 210	Note 6	Magenta	Green	Normal
D3	211		Alt 211	Note 6	Magenta	Cyan	Normal
D4	212		Alt 212	Note 6	Magenta	Red	Normal
D5	213		Alt 213	Note 6	Magenta	Magenta	Normal
D6	214		Alt 214	Note 6	Magenta	Brown	Normal
D7	215		Alt 215	Note 6	Magenta	Light Grey	Normal
D8	216		Alt 216	Note 6	Magenta	Dark Grey	High Intensity
D9	217		Alt 217	Note 6	Magenta	Light Blue	High Intensity Underline
DA	218		Alt 218	Note 6	Magenta	Light Green	High Intensity
DB	219		Alt 219	Note 6	Magenta	Light Cyan	High Intensity
DC	220		Alt 220	Note 6	Magenta	Light Red	High Intensity
DD	221		Alt 221	Note 6	Magenta	Light Magenta	High Intensity
DE	222		Alt 222	Note 6	Magenta	Yellow	High Intensity
DF	223		Alt 223	Note 6	Magenta	White	High Intensity
E0	224	α	Alt 224	Note 6	Yellow	Black	Normal
E1	225	β	Alt 225	Note 6	Yellow	Blue	Underline
E2	226	Γ	Alt 226	Note 6	Yellow	Green	Normal
E3	227	π	Alt 227	Note 6	Yellow	Cyan	Normal
E4	228	Σ	Alt 228	Note 6	Yellow	Red	Normal
E5	229	σ	Alt 229	Note 6	Yellow	Magenta	Normal
E6	230	μ	Alt 230	Note 6	Yellow	Brown	Normal
E7	231	τ	Alt 231	Note 6	Yellow	Light Grey	Normal
E8	232	Φ	Alt 232	Note 6	Yellow	Dark Grey	High Intensity
E9	233	θ	Alt 233	Note 6	Yellow	Light Blue	High Intensity Underline
EA	234	Ω	Alt 234	Note 6	Yellow	Light Green	High Intensity
EB	235	δ	Alt 235	Note 6	Yellow	Light Cyan	High Intensity

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
EC	236	∞	Alt 236	Note 6	Yellow	Light Red	High Intensity
ED	237	φ	Alt 237	Note 6	Yellow	Light Magenta	High Intensity
EE	238	ε	Alt 238	Note 6	Yellow	Yellow	High Intensity
EF	239	∩	Alt 239	Note 6	Yellow	White	High Intensity
F0	240	≡	Alt 240	Note 6	White	Black	Reverse Video
F1	241	±	Alt 241	Note 6	White	Blue	Underline
F2	242	≥	Alt 242	Note 6	White	Green	Normal
F3	243	≤	Alt 243	Note 6	White	Cyan	Normal
F4	244	⌠	Alt 244	Note 6	White	Red	Normal
F5	245	⌡	Alt 245	Note 6	White	Magenta	Normal
F6	246	÷	Alt 246	Note 6	White	Brown	Normal
F7	247	≈	Alt 247	Note 6	White	Light Grey	Normal
F8	248	○	Alt 248	Note 6	White	Dark Grey	Reverse Video
F9	249	●	Alt 249	Note 6	White	Light Blue	High Intensity Underline
FA	250	·	Alt 250	Note 6	White	Light Green	High Intensity
FB	251	√	Alt 251	Note 6	White	Light Cyan	High Intensity
FC	252	η	Alt 252	Note 6	White	Light Red	High Intensity
FD	253	2	Alt 253	Note 6	White	Light Magenta	High Intensity
FE	254	■	Alt 254	Note 6	White	Yellow	High Intensity
FF	255	**BLANK**	Alt 255	Note 6	White	White	High Intensity

NOTE 1 Asterisk (*) can easily be keyed using two methods: 1) hit the Prt Sc key or 2) in shift mode hit the 8 key.

NOTE 2 Period (.) can easily be keyed using two methods: 1) hit the > key or 2) in shift or Num Lock mode hit the Del key.

NOTE 3 Numeric characters (0—9) can easily be keyed using two methods: 1) hit the numeric keys on the top row of the typewriter portion of the keyboard or 2) in shift or Num Lock mode hit the numeric keys in the 10—key pad portion of the keyboard.

NOTE 4 Upper case alphabetic characters (A—Z) can easily be keyed in two modes: 1) in shift mode the appropriate alphabetic key or 2) in Caps Lock mode hit the appropriate alphabetic key.

NOTE 5 Lower case alphabetic characters (a—z) can easily be keyed in two modes: 1) in "normal" mode hit the appropriate key or 2) in Caps Lock combined with shift mode hit the appropriate alphabetic key.

NOTE 6 The 3 digits after the Alt key must be typed from the numeric key pad (keys 71—73, 75—77, 79—82). Character codes 000 through 255 can be entered in this fashion. (With Caps Lock activated, Character codes 97 through 122 will display upper case rather than lower case alphabetic characters.)

Character Set (00-7F) Quick Reference

DECIMAL VALUE ➡		0	16	32	48	64	80	96	112
⬇	HEXA DECIMAL VALUE	0	1	2	3	4	5	6	7
0	0	BLANK (NULL)	►	BLANK (SPACE)	0	@	P	‘	p
1	1	☺	◄	!	1	A	Q	a	q
2	2	☻	↕	"	2	B	R	b	r
3	3	♥	!!	#	3	C	S	c	s
4	4	♦	¶	$	4	D	T	d	t
5	5	♣	§	%	5	E	U	e	u
6	6	♠	▬	&	6	F	V	f	v
7	7	•	↨	’	7	G	W	g	w
8	8	◘	↑	(	8	H	X	h	x
9	9	○	↓	)	9	I	Y	i	y
10	A	◙	→	*	:	J	Z	j	z
11	B	♂	←	+	;	K	[	k	{
12	C	♀	∟	,	<	L	\	l	¦
13	D	♪	↔	—	=	M	]	m	}
14	E	♫	▲	.	>	N	^	n	~
15	F	☼	▼	/	?	O	_	o	△

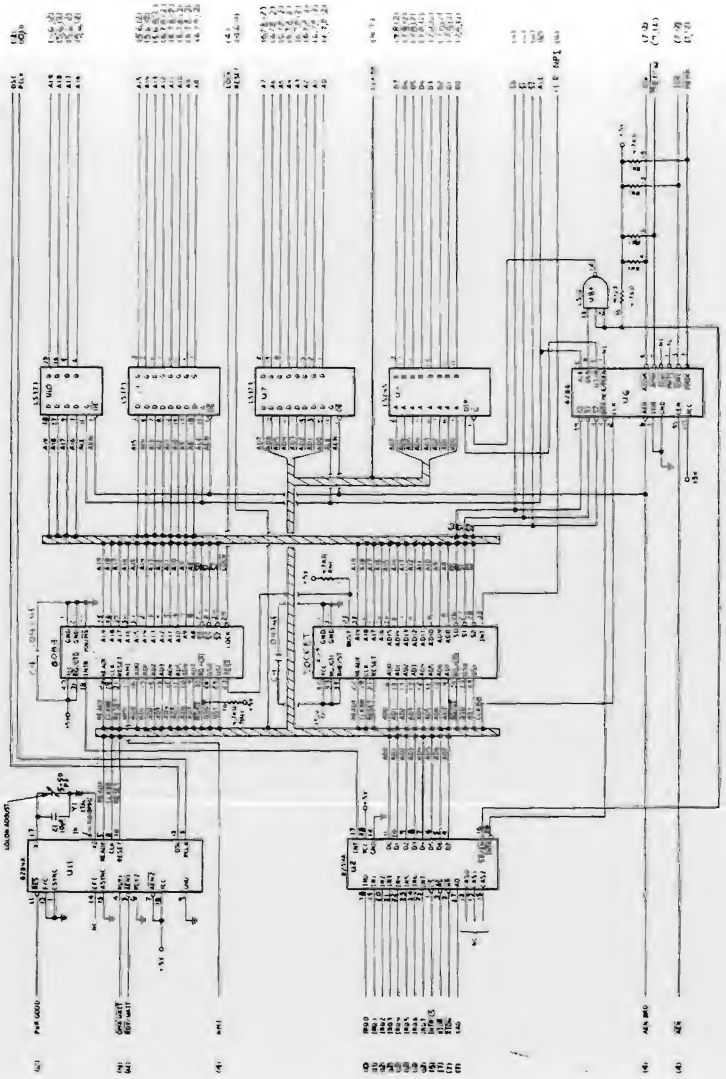
Character Set (80-FF) Quick Reference

DECIMAL VALUE ➡		128	144	160	176	192	208	224	240
⬇	HEXA DECIMAL VALUE	8	9	A	B	C	D	E	F
0	0	Ç	É	á	▒	└	╨	∝	≡
1	1	ü	æ	í	▒	┴	╤	β	±
2	2	é	Æ	ó	▓	┬	╥	Γ	≥
3	3	â	ô	ú	│	├	╙	π	≤
4	4	ä	ö	ñ	┤	─	╘	Σ	⌠
5	5	à	ò	Ñ	╡	┼	╒	σ	⌡
6	6	å	û	ª	╢	╞	╓	μ	÷
7	7	ç	ù	º	╖	╟	╫	τ	≈
8	8	ê	ÿ	¿	╕	╚	╪	Φ	°
9	9	ë	Ö	⌐	╣	╔	┘	Θ	•
10	A	è	Ü	¬	║	╩	┌	Ω	•
11	B	ï	¢	½	╗	╦	■	δ	√
12	C	î	£	¼	╝	╠	▄	∞	n
13	D	ì	¥	¡	╜	═	▌	φ	2
14	E	Ä	₧	«	╛	╬	▐	∈	■
15	F	Å	ƒ	»	┐	╧	▀	∩	BLANK 'FF'

NOTES

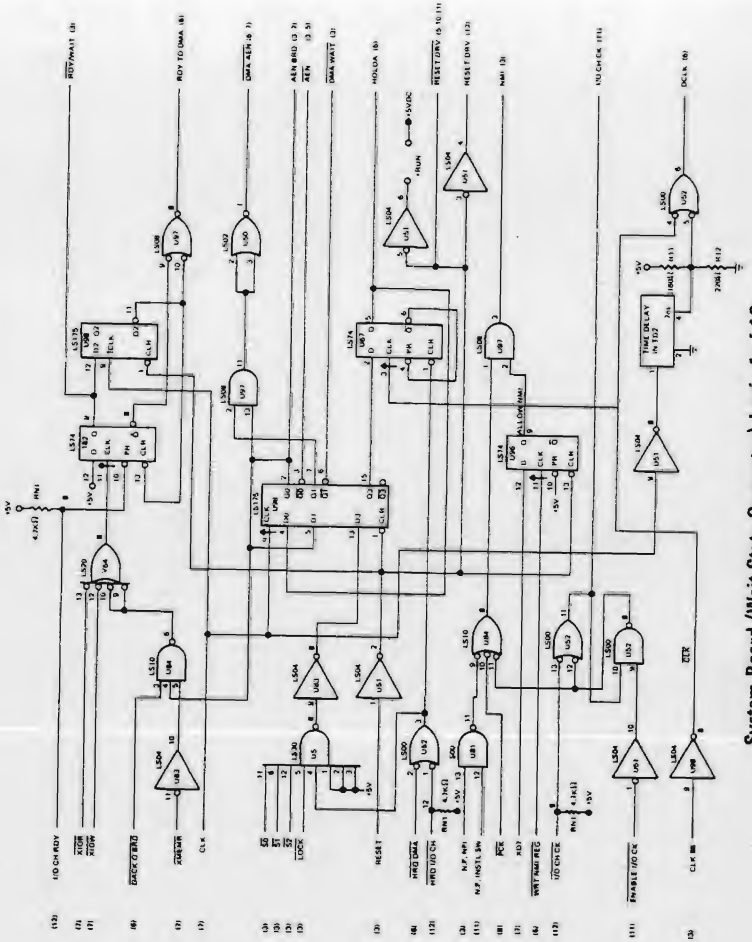
APPENDIX D. LOGIC DIAGRAMS

Contents

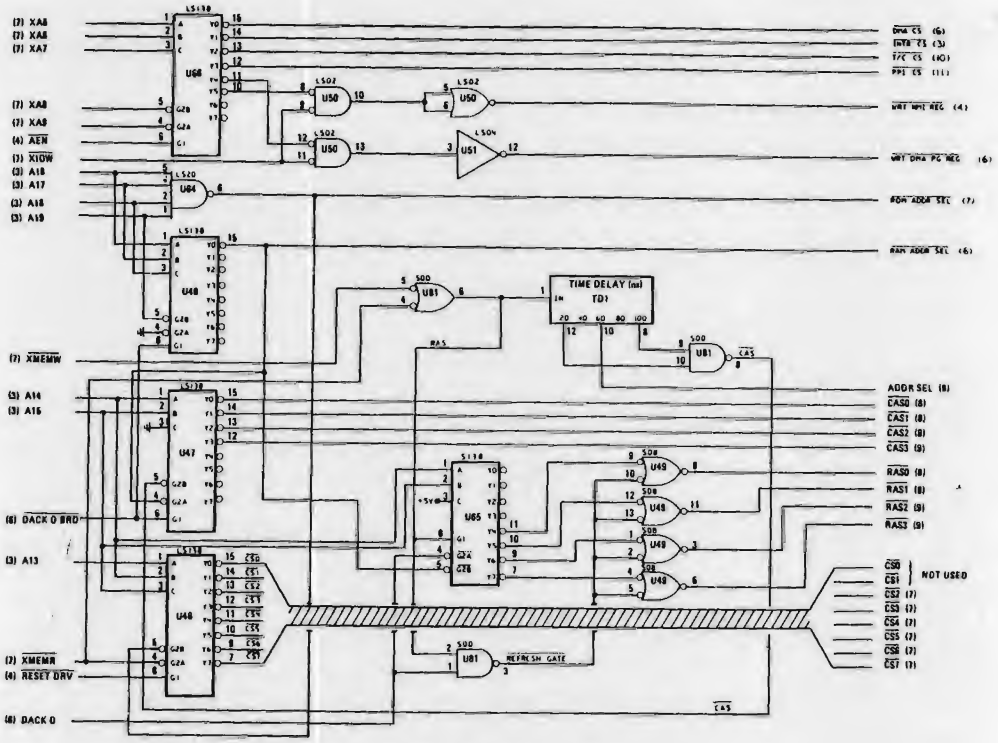


System Board (Processor And Support) Logic 3 of 12

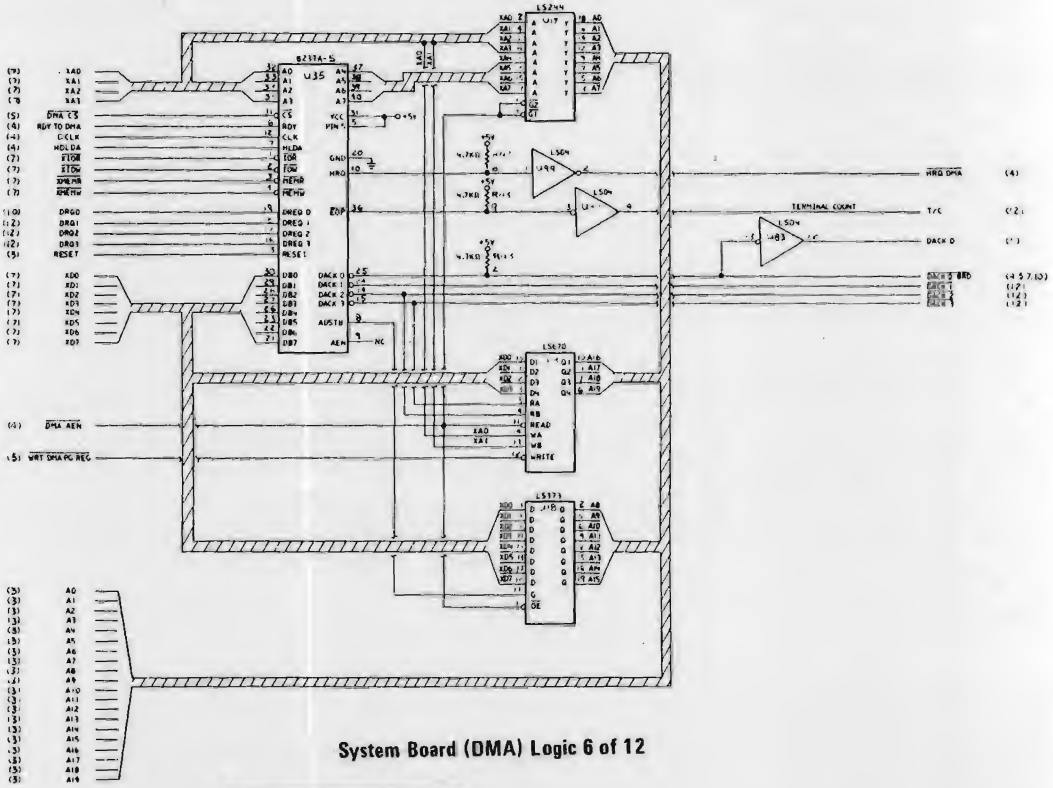
Note: Logics one and two of twelve are not applicable



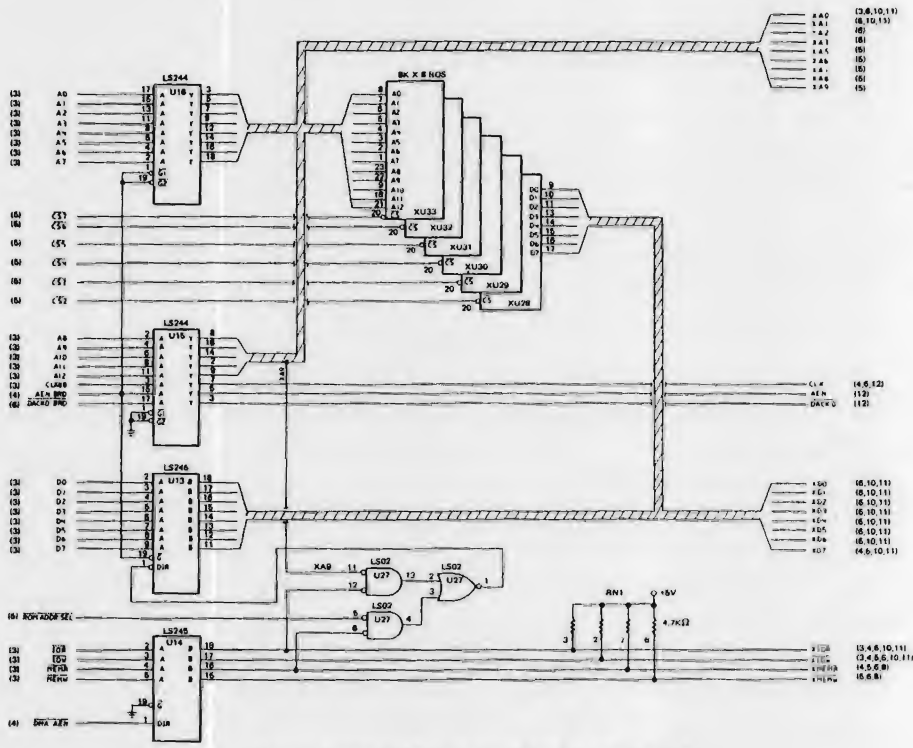
System Board (Wait State Generator) Logic 4 of 12



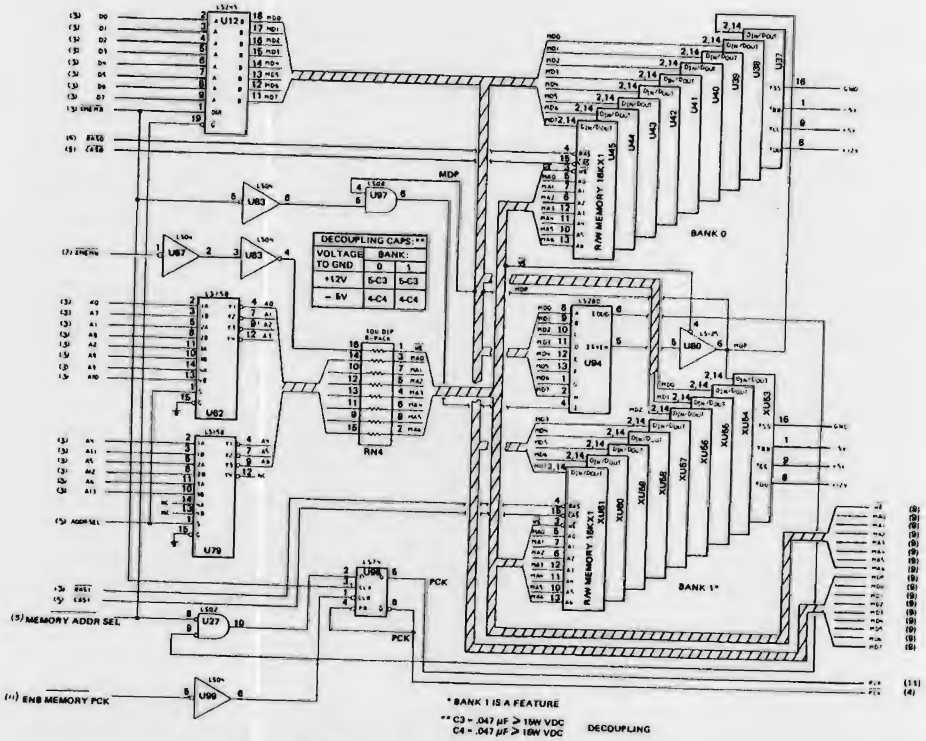
System Board (Device Decodes) Logic 5 of 12



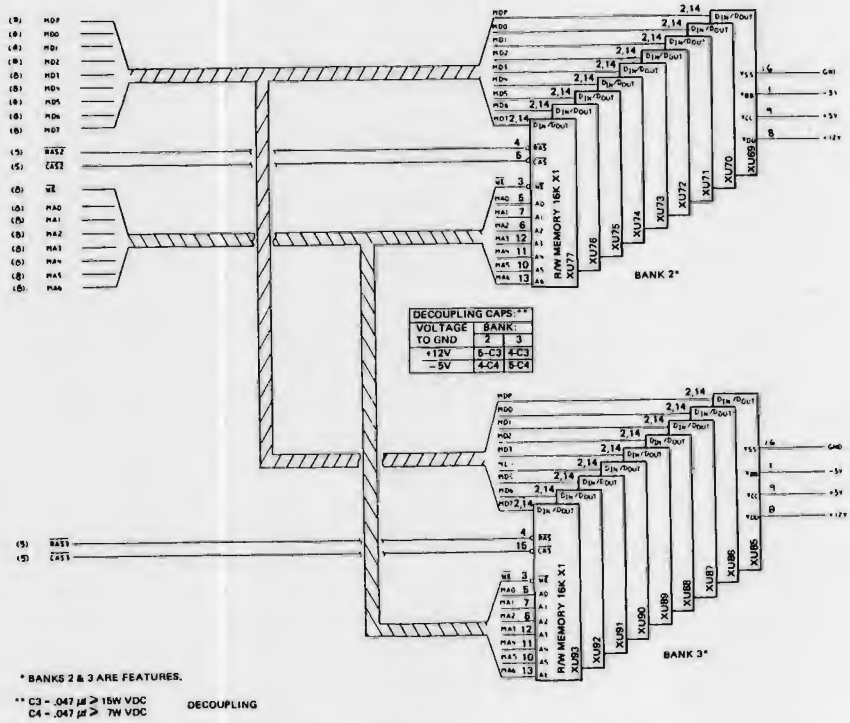
System Board (DMA) Logic 6 of 12



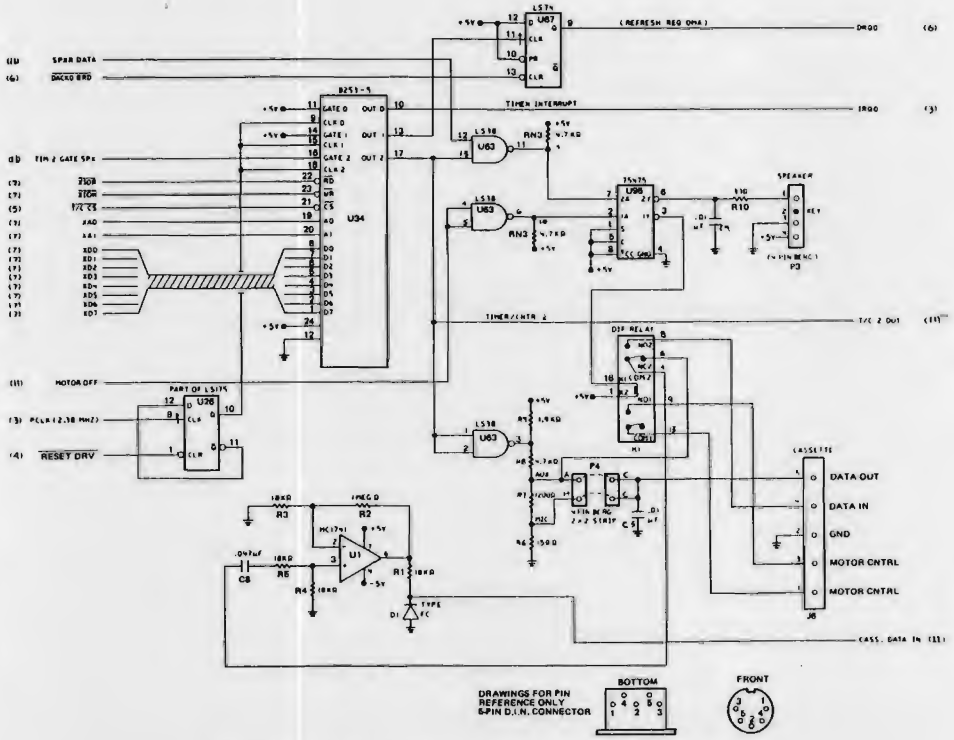
System Board (ROS And Bus Driver) Logic 7 of 12



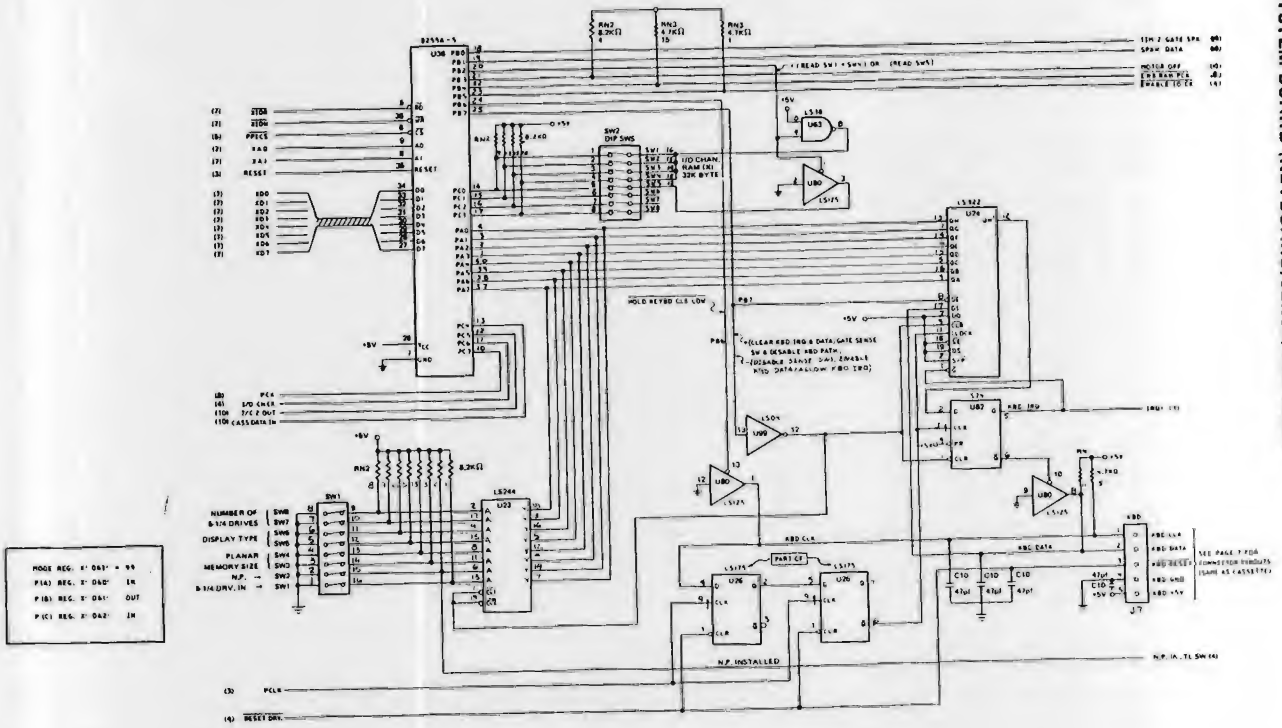
System Board (Dynamic Memory) Logic 8 of 12



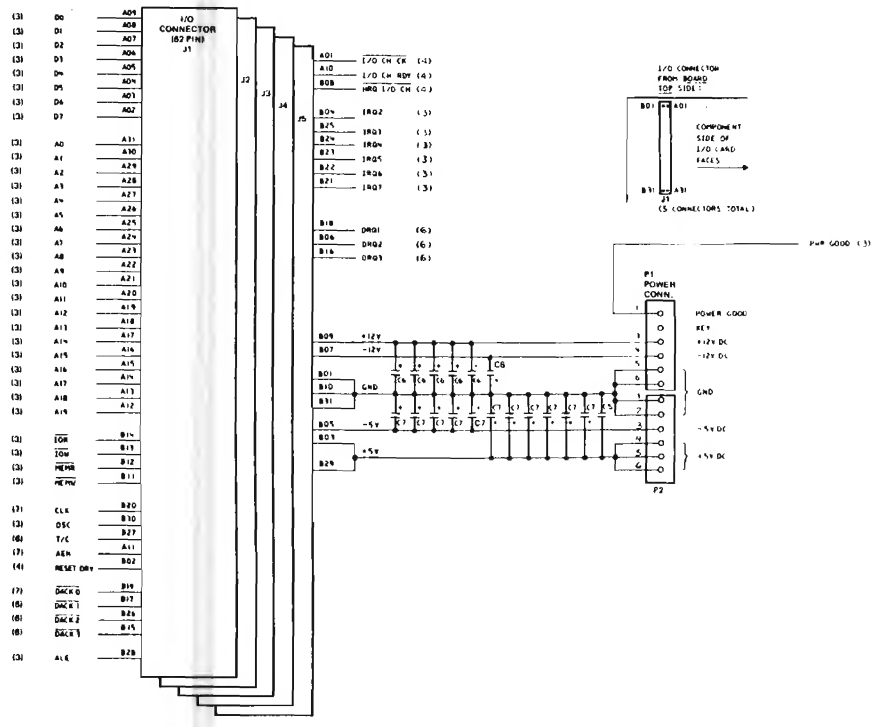
System Board (Dynamic Memory Extended) Logic 9 of 12



System Board (Speaker/Cassette/Timer/Counter) Logic 10 of 12

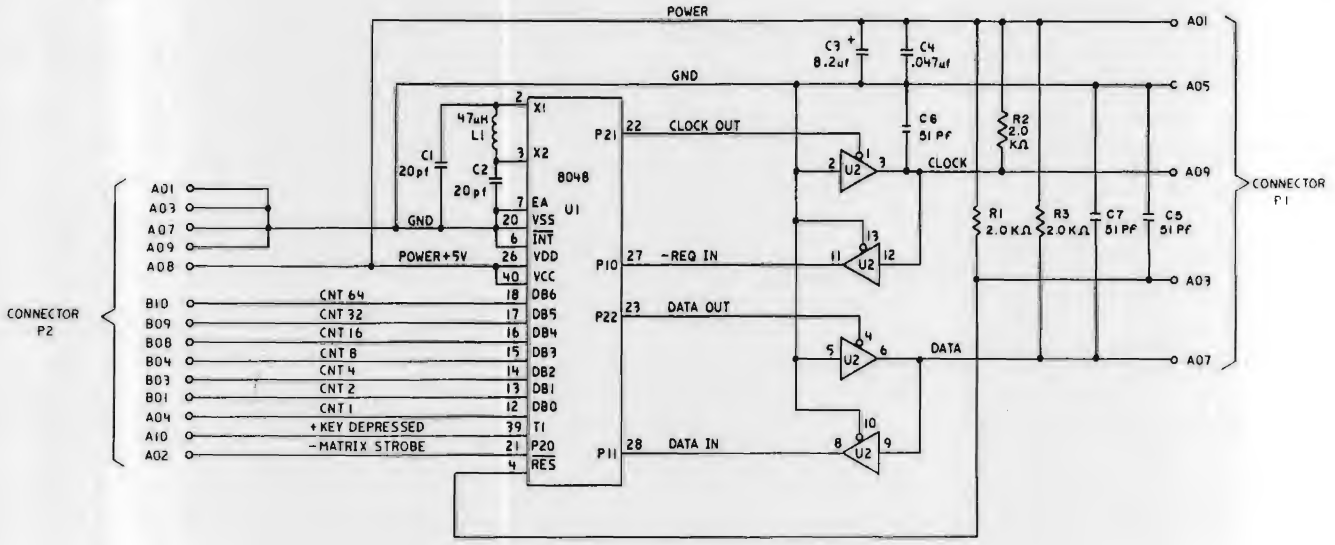


System Board (Keyboard/Sense/Control) Logic 11 of 12

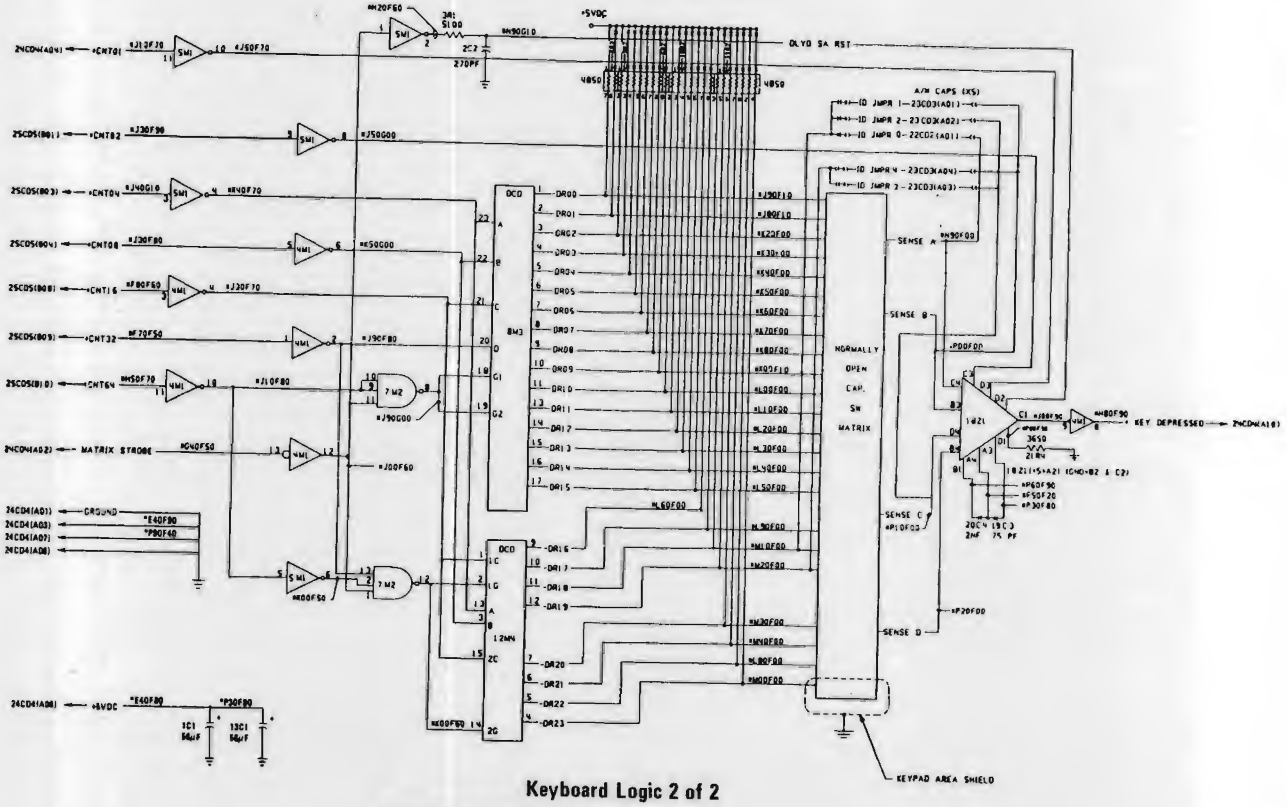


NOTE:
ALL CAPS ARE 8.2μF TANTALUM ON THIS PAGE

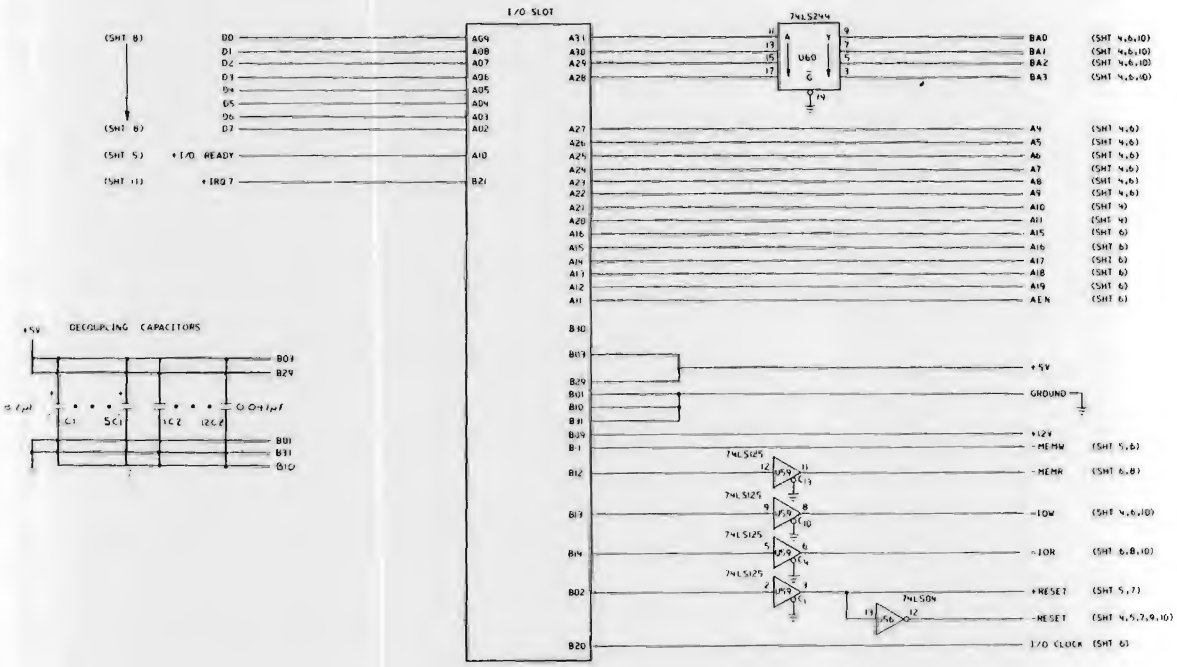
System Board (I/O Channel) Logic 12 of 12



Keyboard Logic 1 of 2

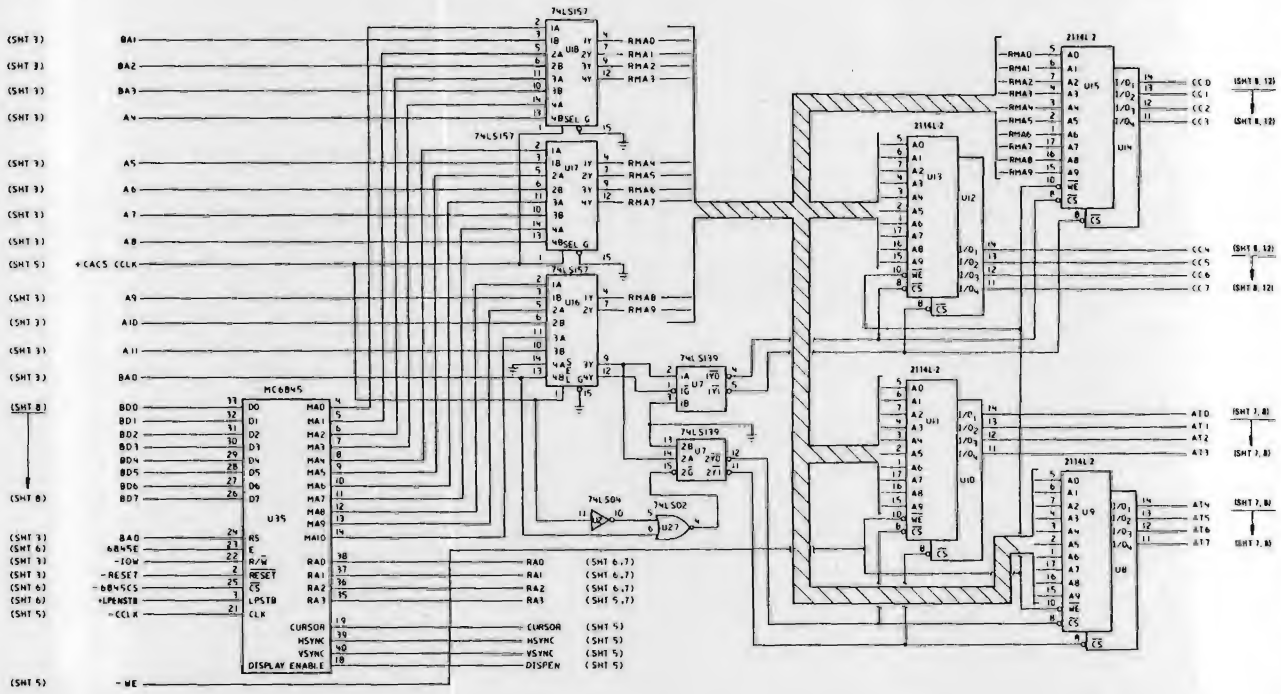


Keyboard Logic 2 of 2

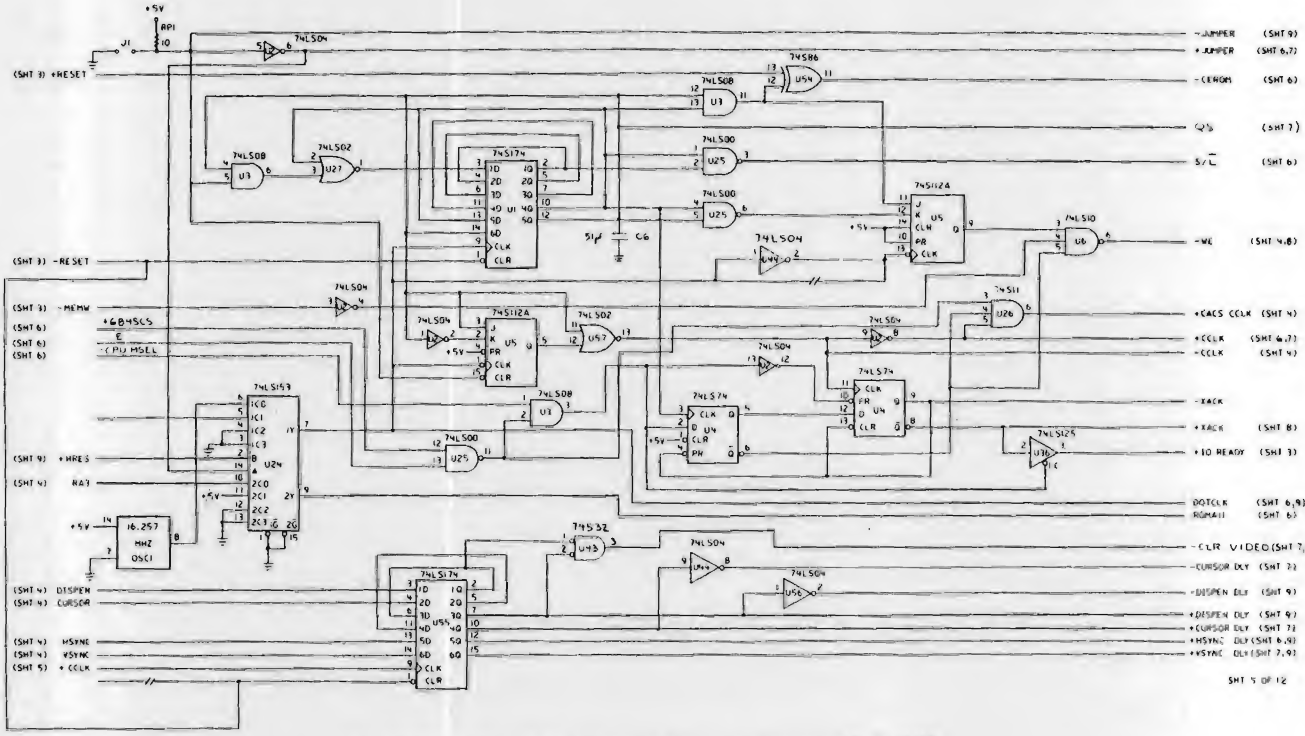


Note: Logics one and two of twelve are not applicable.

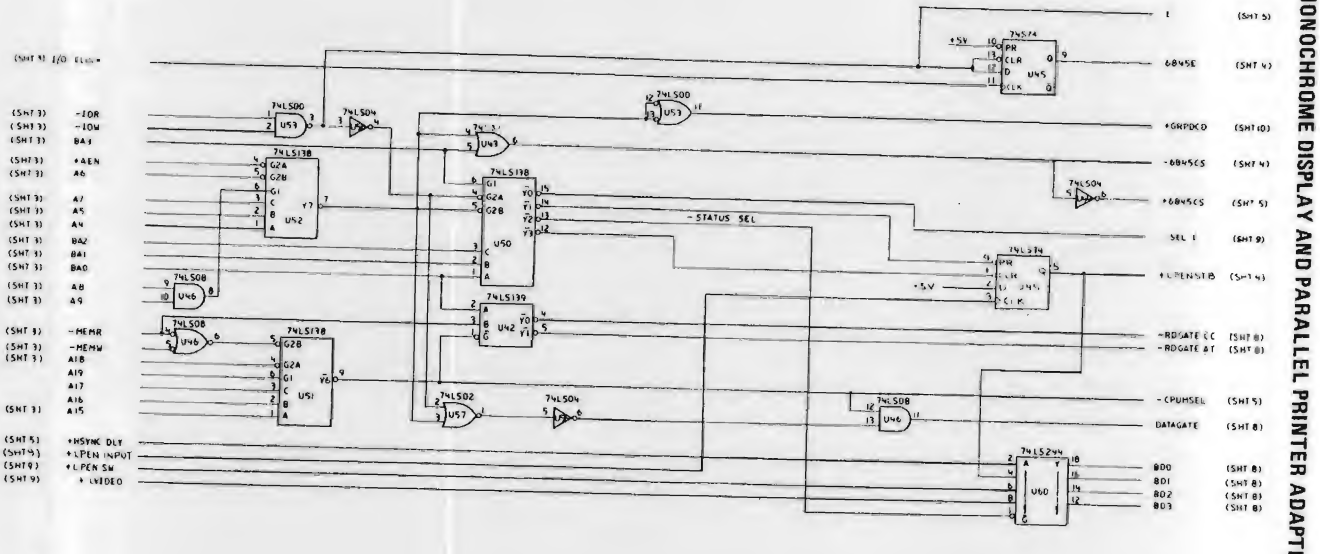
IBM Monochrome Display And Parallel Printer Adapter Logic 3 of 12



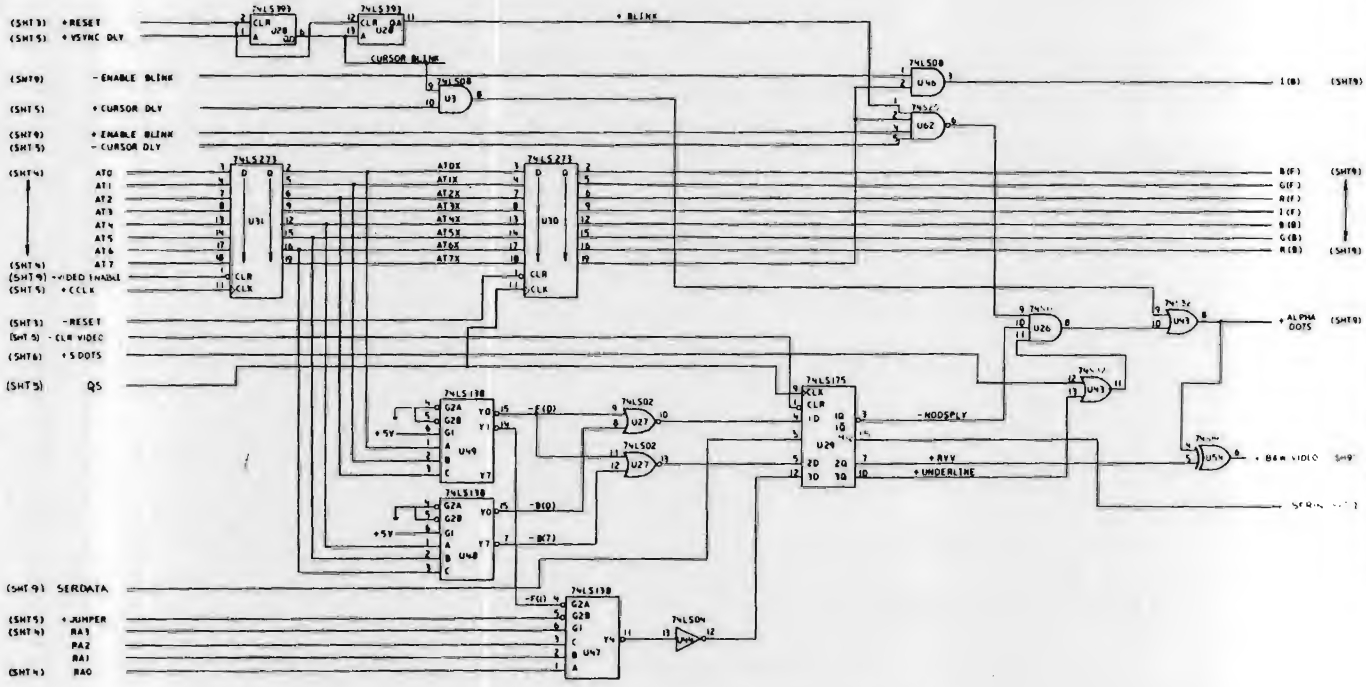
IBM Monochrome Display And Parallel Printer Adapter Logic 4 of 12



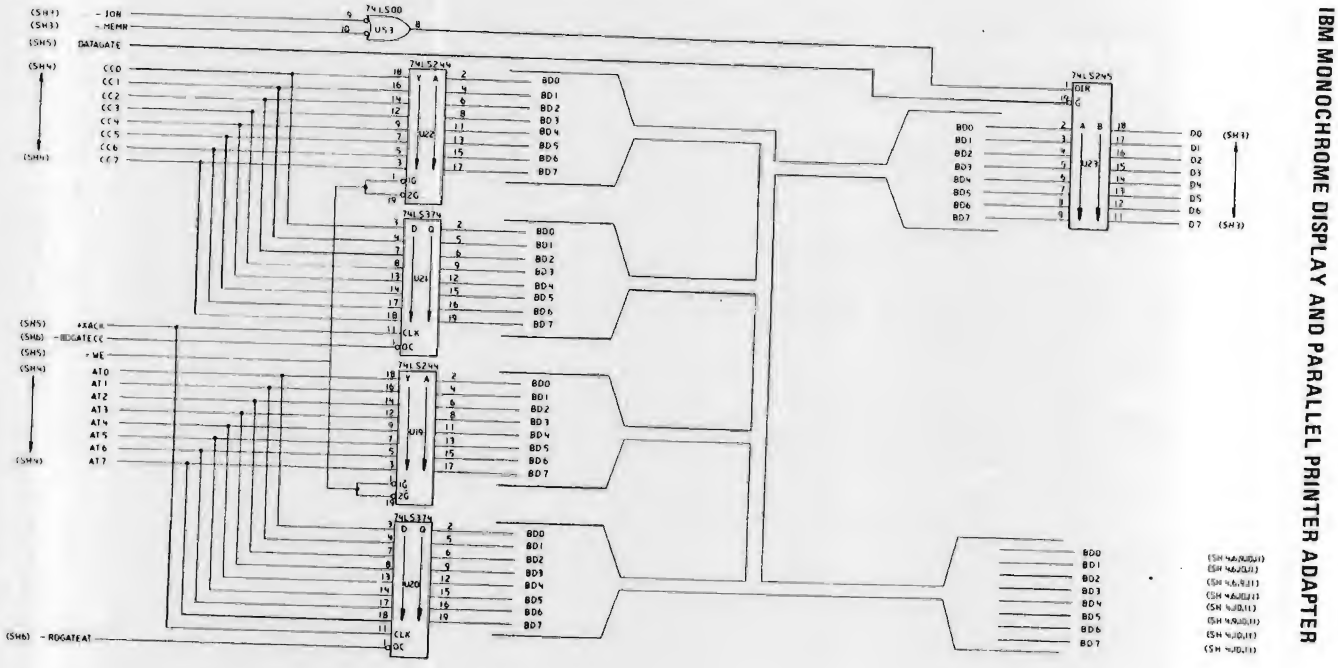
IBM Monochrome Display And Parallel Printer Adapter Logic 5 of 12



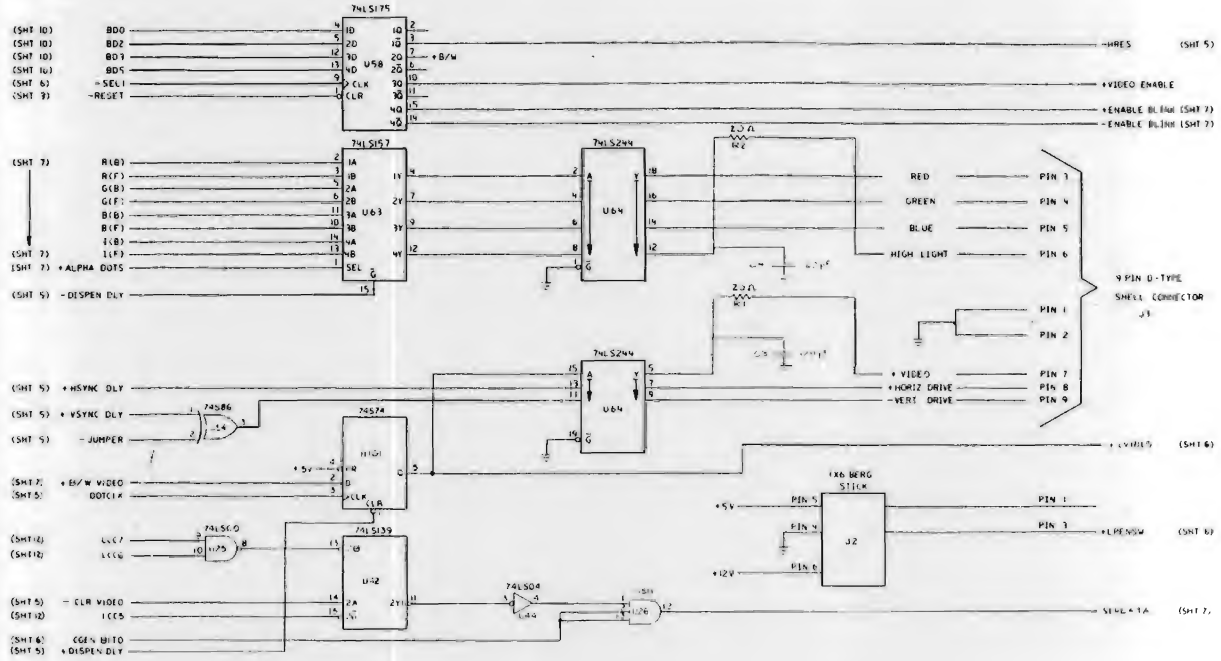
IBM Monochrome Display And Parallel Printer Adapter Logic 6 of 12



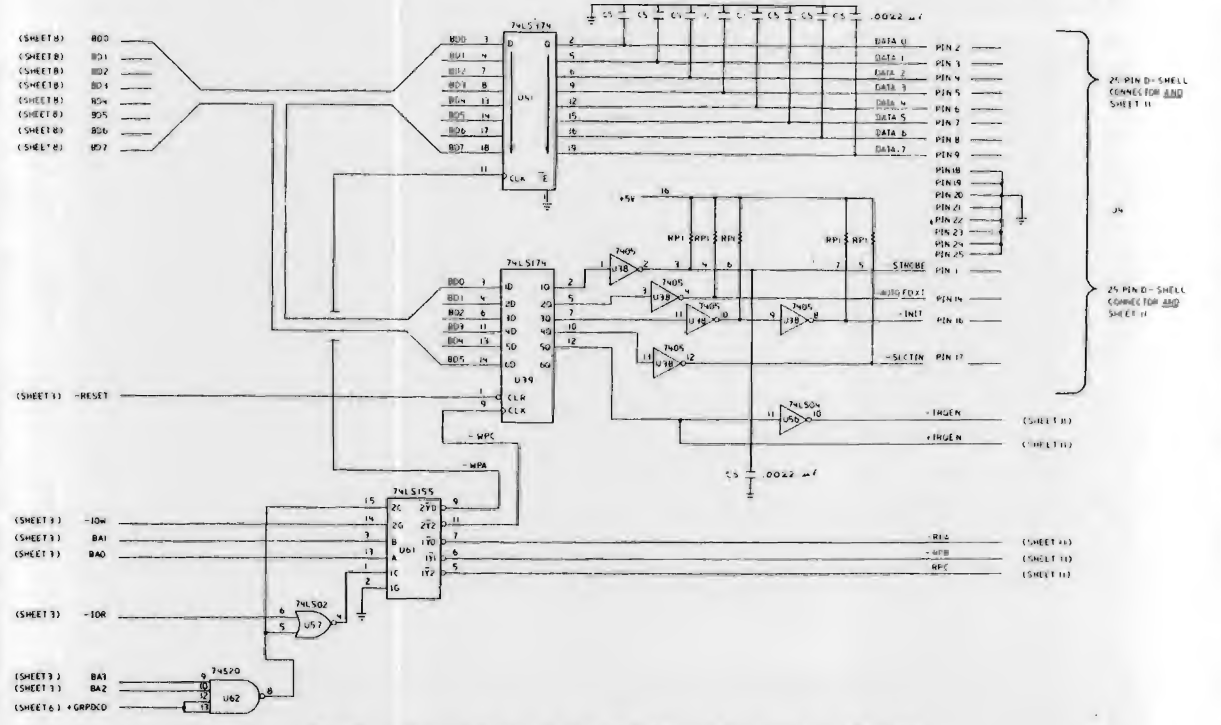
IBM Monochrome Display And Parallel Printer Adapter Logic 7 of 12



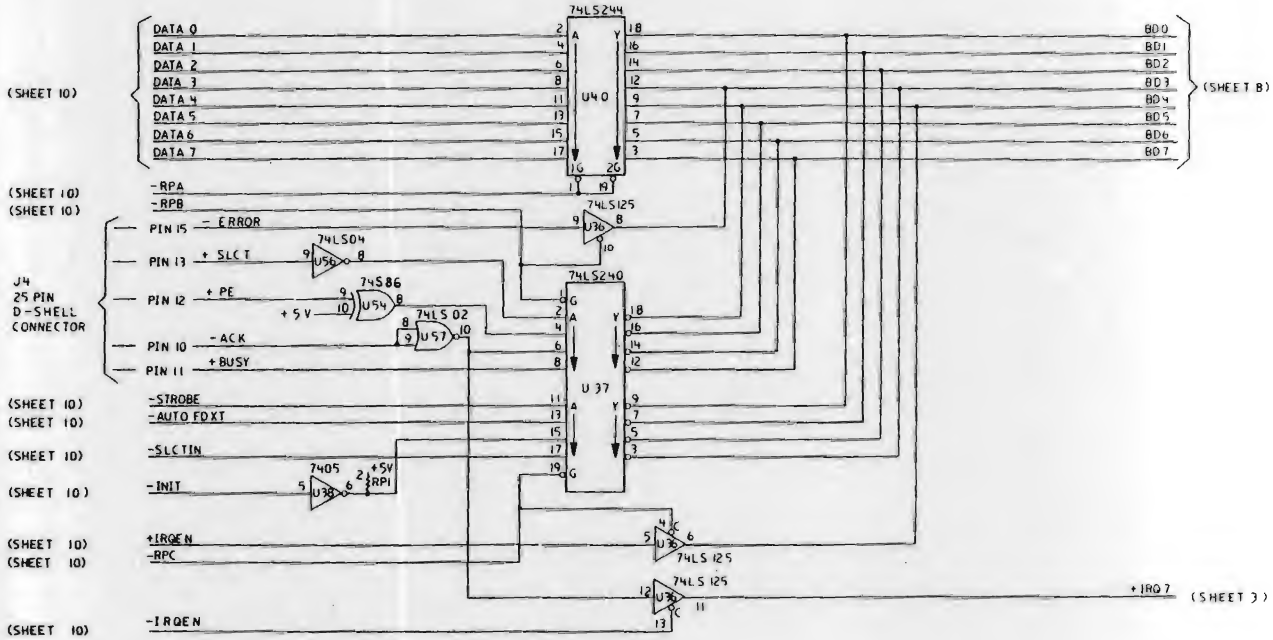
IBM Monochrome Display And Parallel Printer Adapter Logic 8 of 12



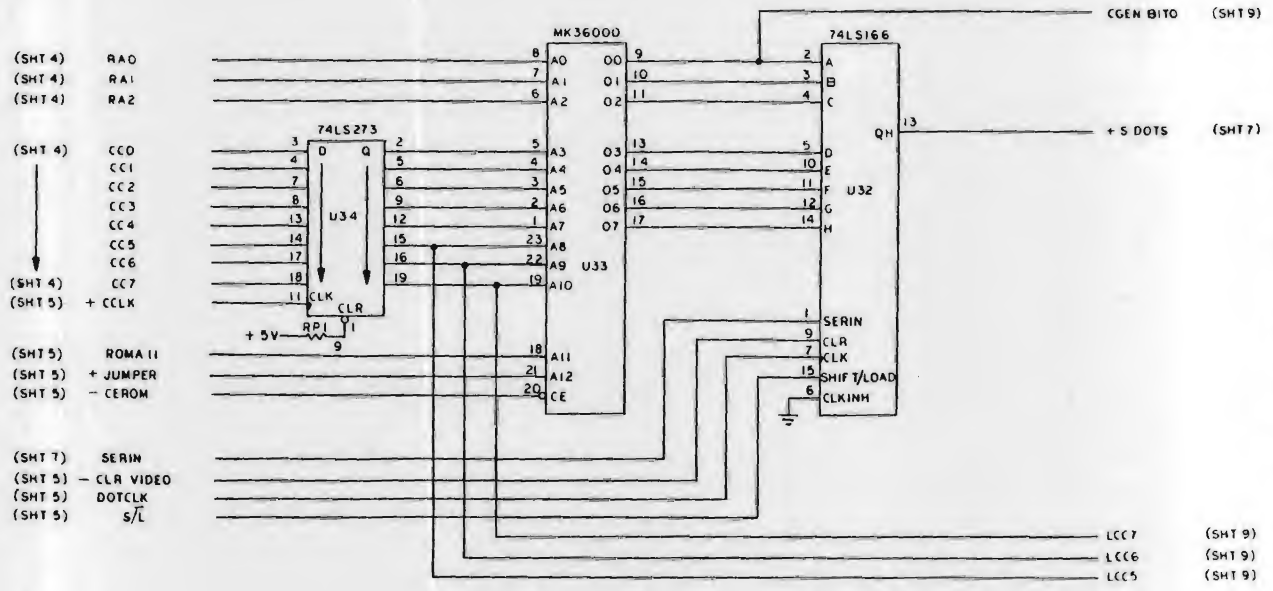
IBM Monochrome Display And Parallel Printer Adapter Logic 9 of 12



IBM Monochrome Display And Parallel Printer Adapter Logic 10 of 12



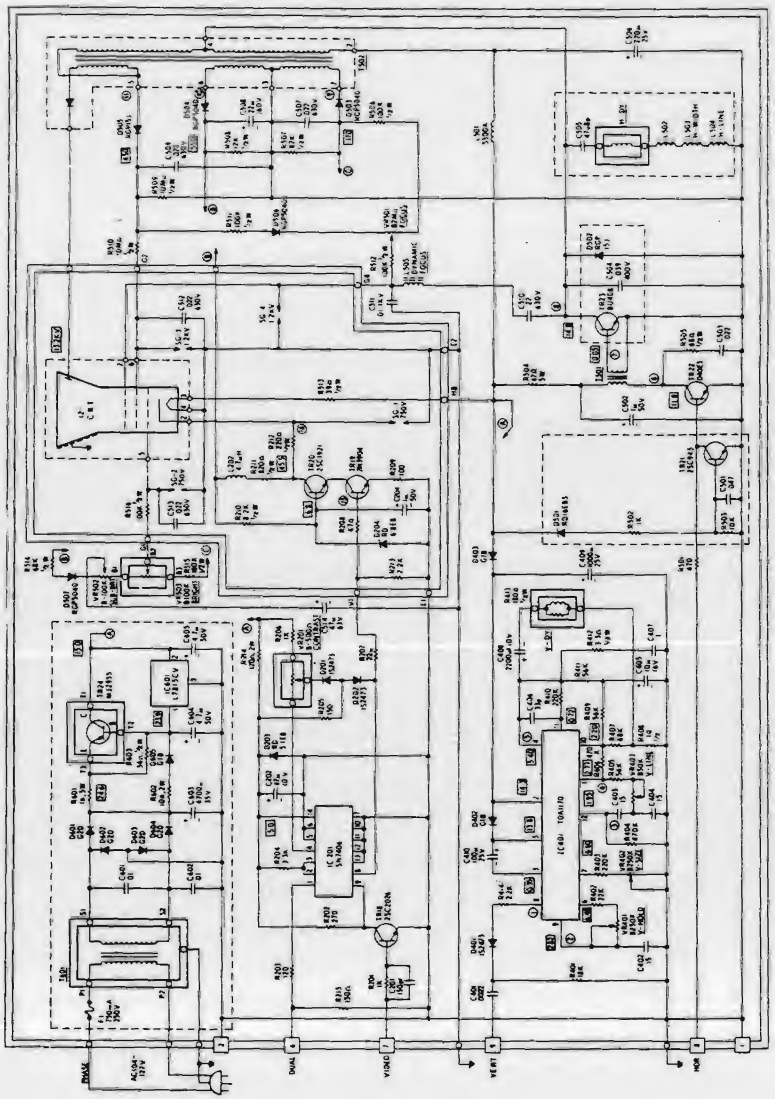
IBM Monochrome Display And Parallel Printer Adapter Logic 11 of 12



IBM Monochrome Display And Parallel Printer Adapter Logic 12 of 12

DANGER

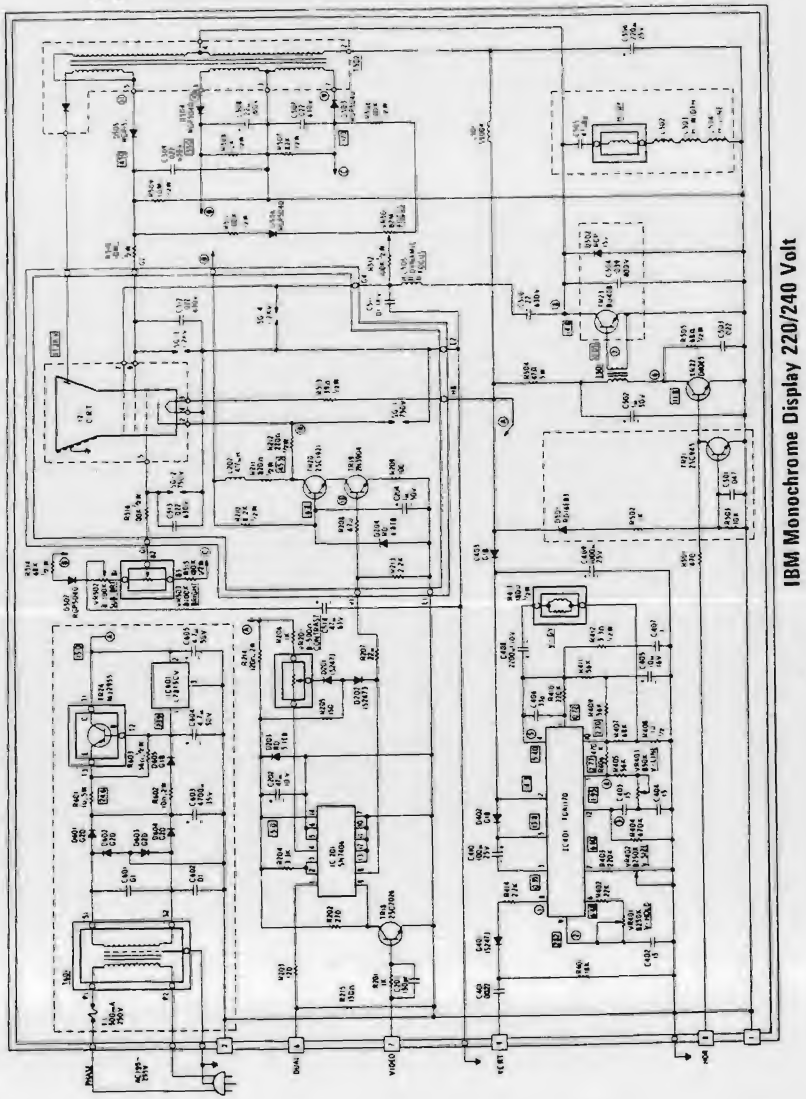
HAZARDOUS VOLTAGES UP TO 450 VOLTS EXIST ON THE PRINTED CIRCUIT BOARDS



IBM Monochrome Display 120 Volt

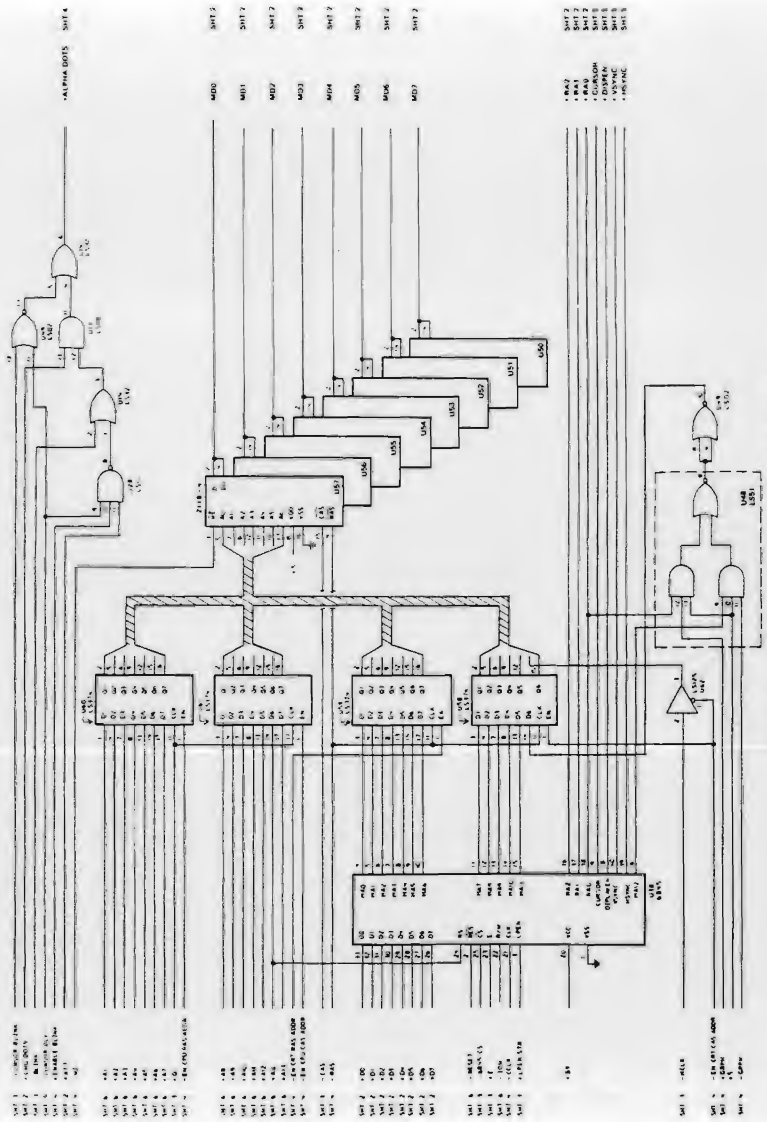
DANGER

HAZARDOUS VOLTAGES UP TO 450 VOLTS EXIST ON THE PRINTED CIRCUIT BOARDS

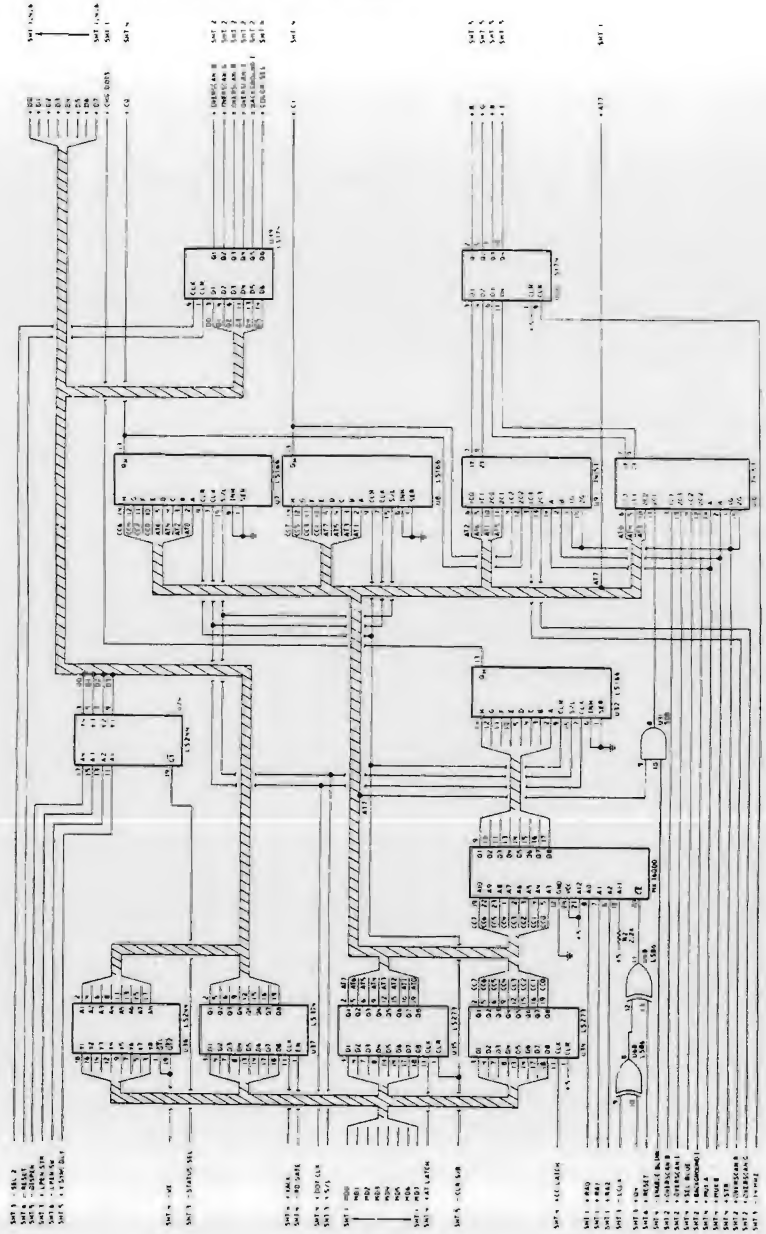


IBM Monochrome Display 220/240 Volt

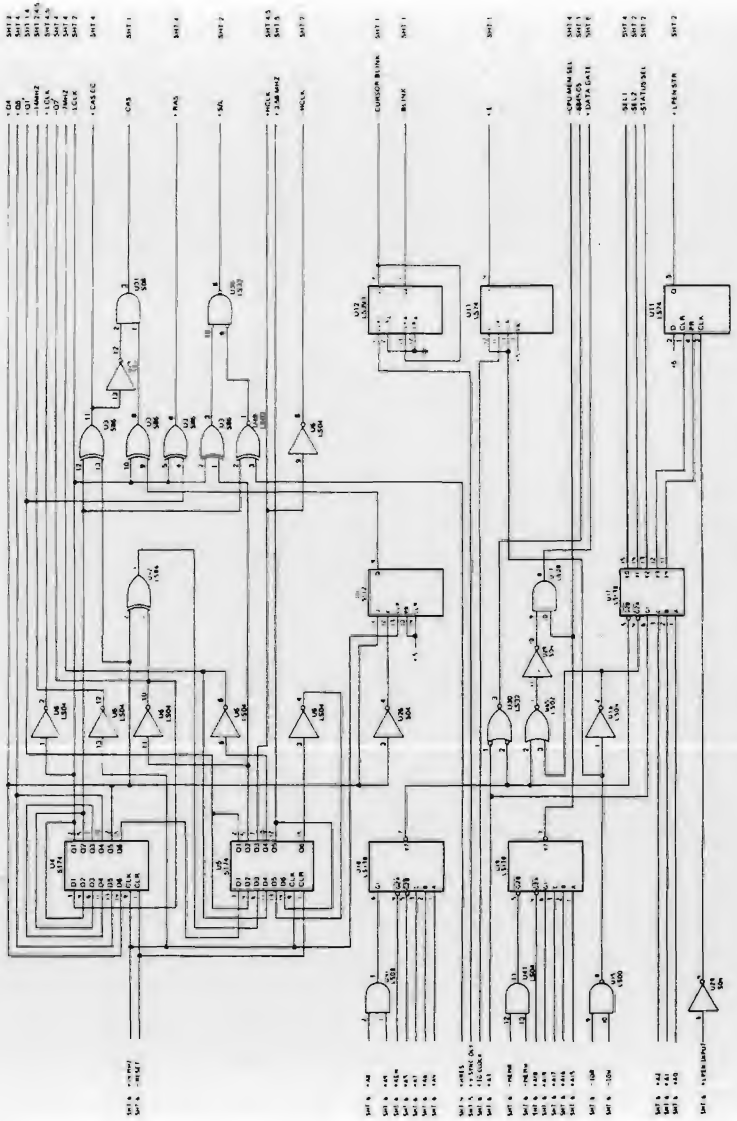
NOTES



Color/Graphics Monitor Adapter Logic 1 of 6

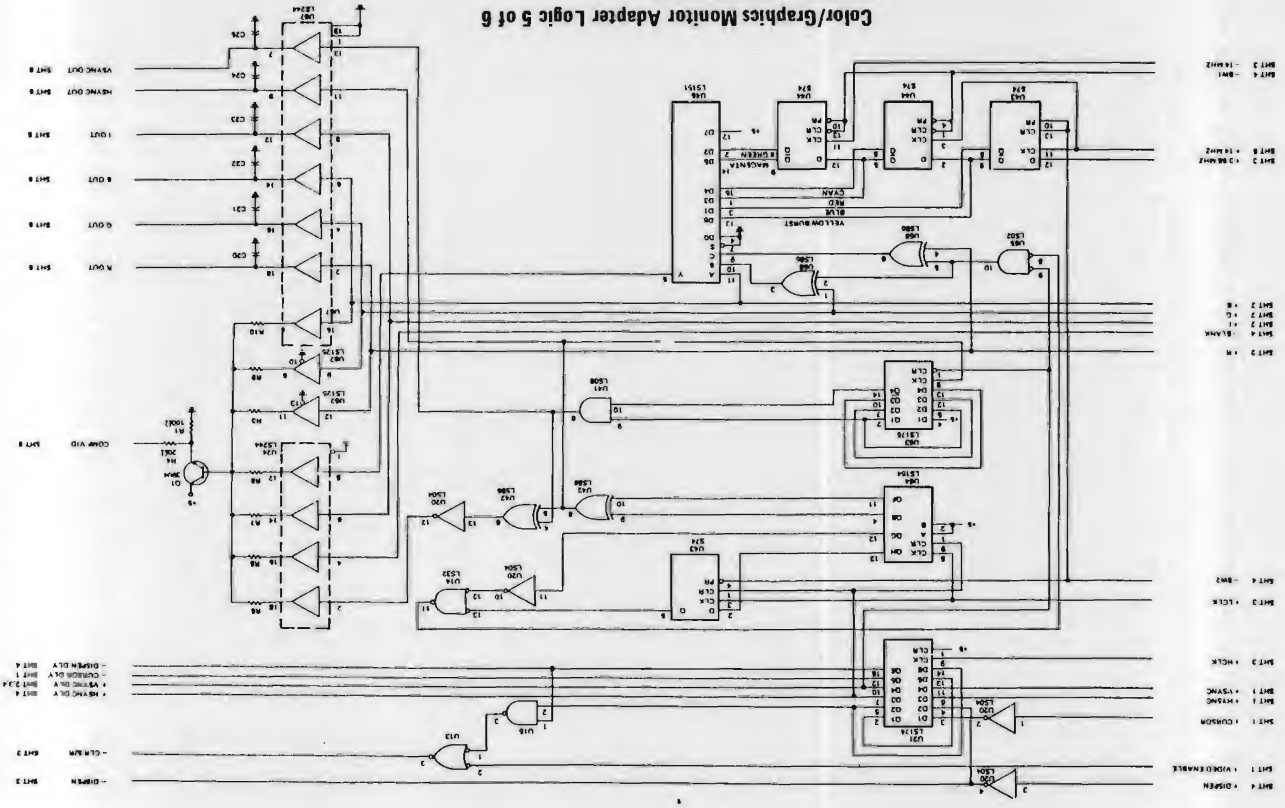


Color/Graphics Monitor Adapter Logic 2 of 6

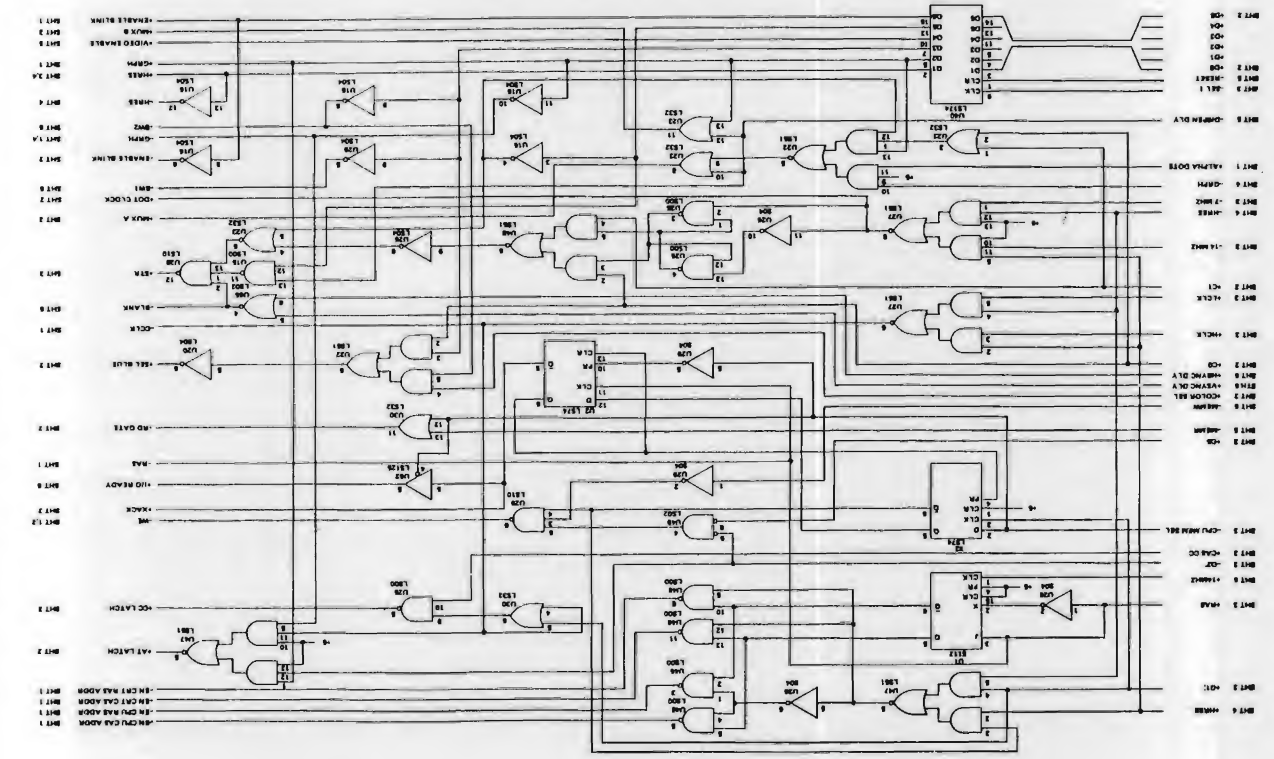


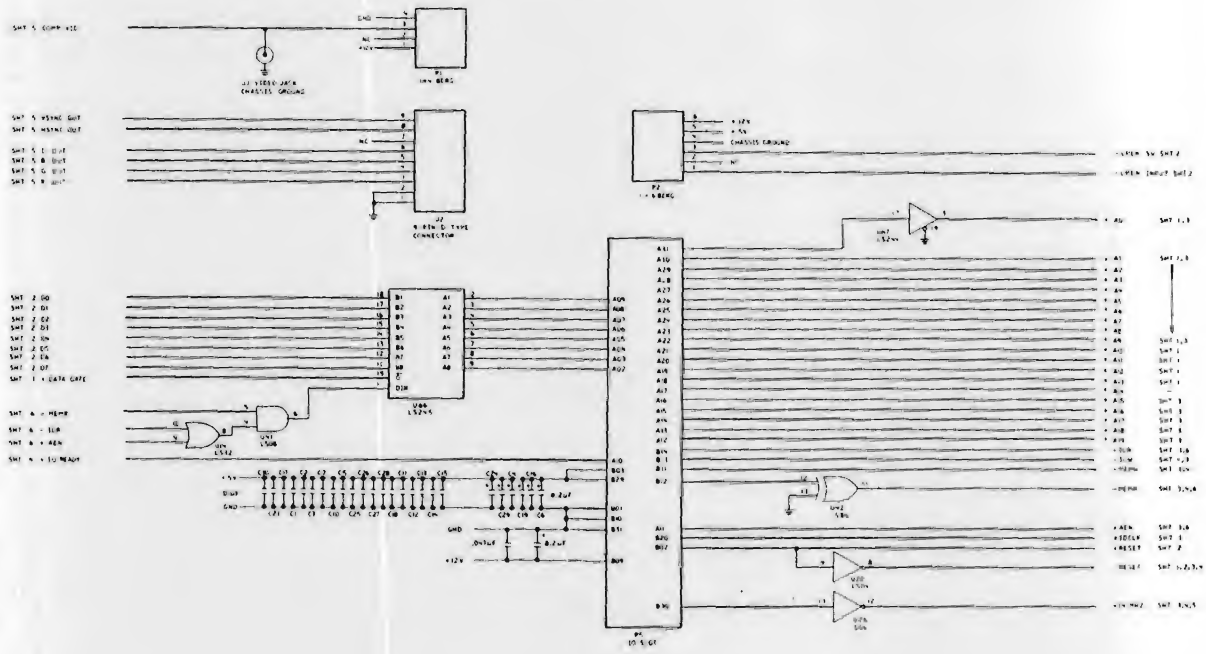
Color/Graphics Monitor Adapter Logic 3 of 6

Color/Graphics Monitor Adapter Logic 5 of 6

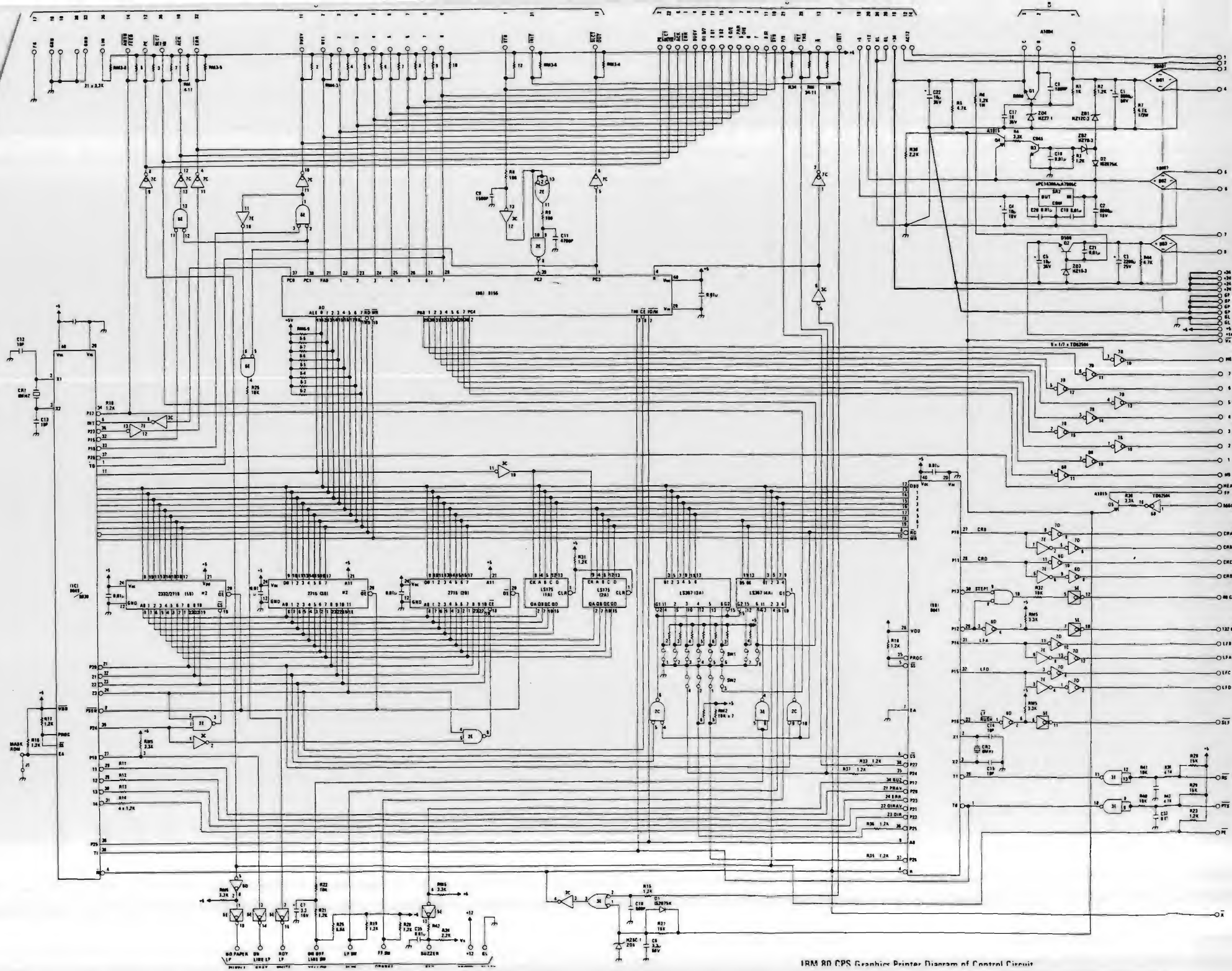


Color/Graphics Monitor Adapter Logic 4 of 6



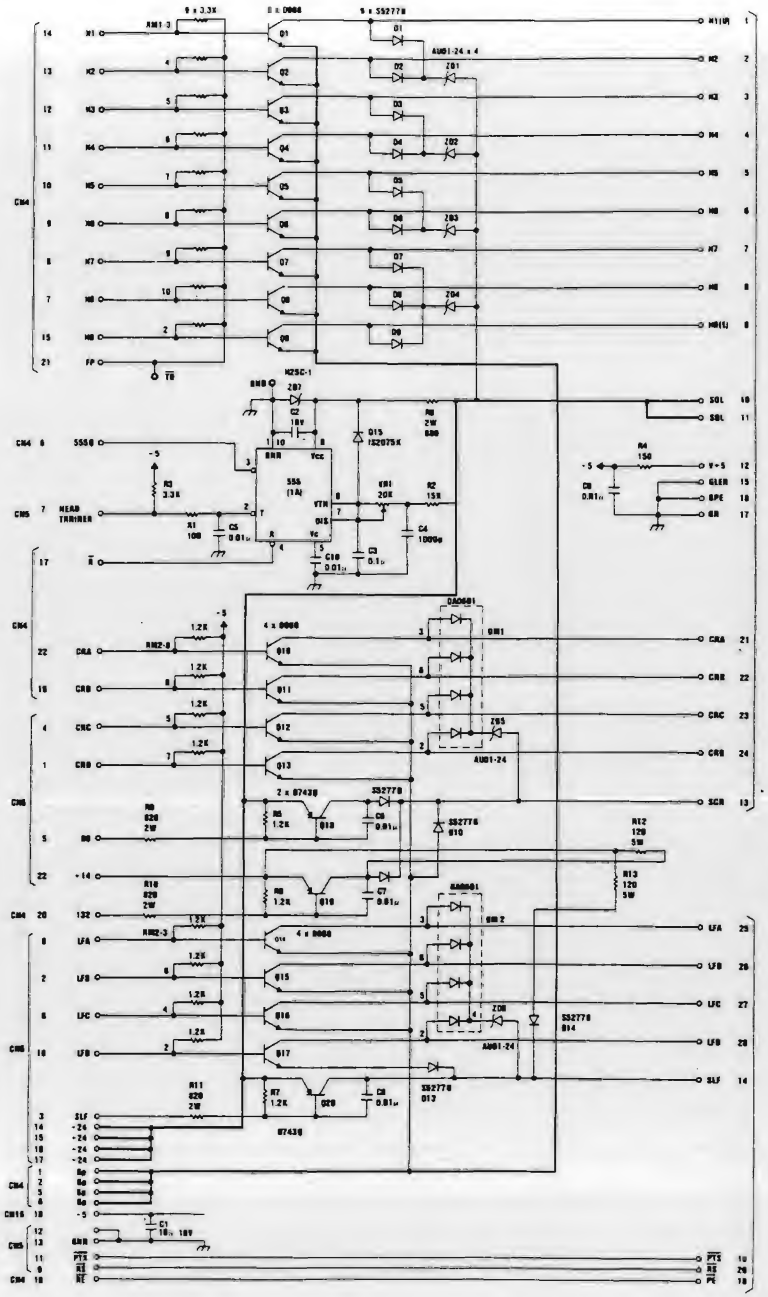


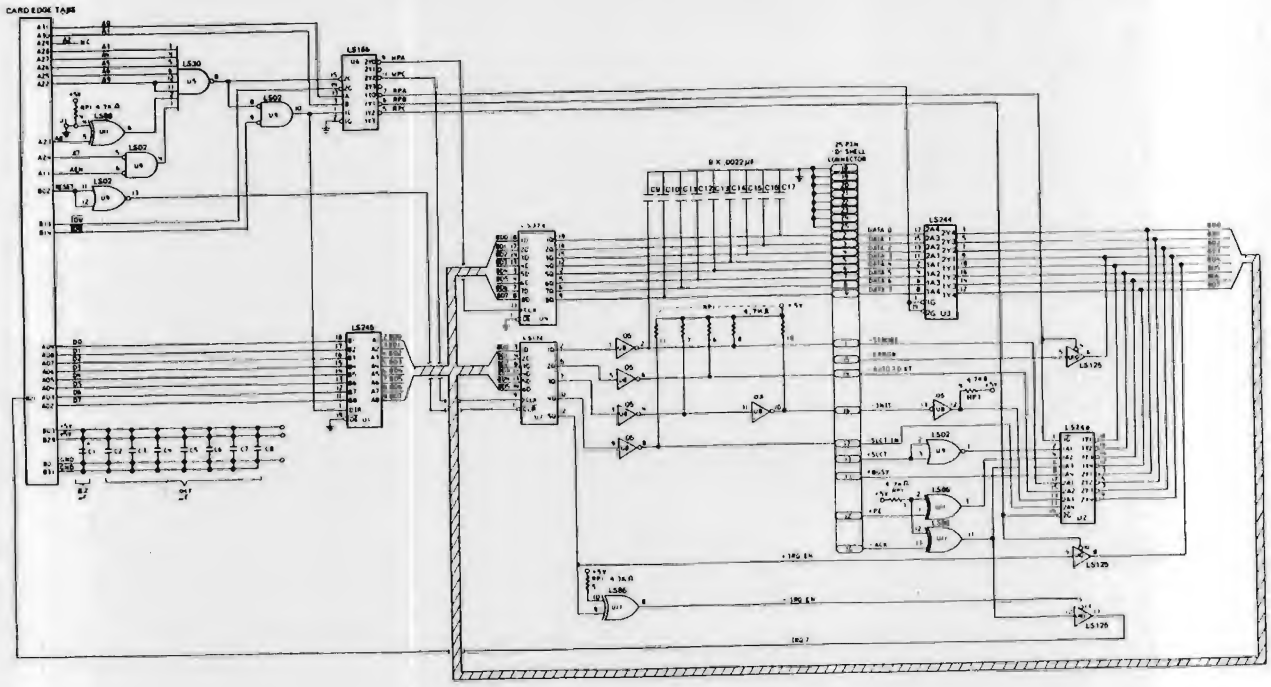
Color/Graphics Monitor Adapter Logic 6 of 6



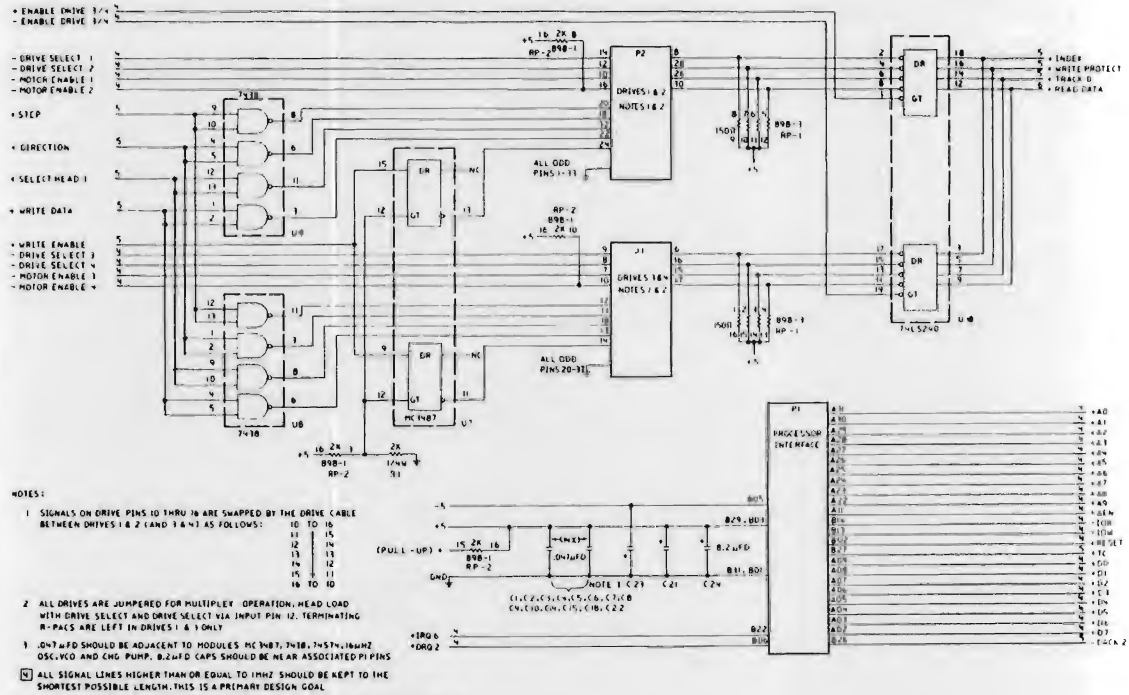
IBM 80 CPS Graphics Printer Diagram of Control Circuit

IBM 80 CPS DIAGRAM OF DRIVER CIRCUIT GRAPHICS PRINTER



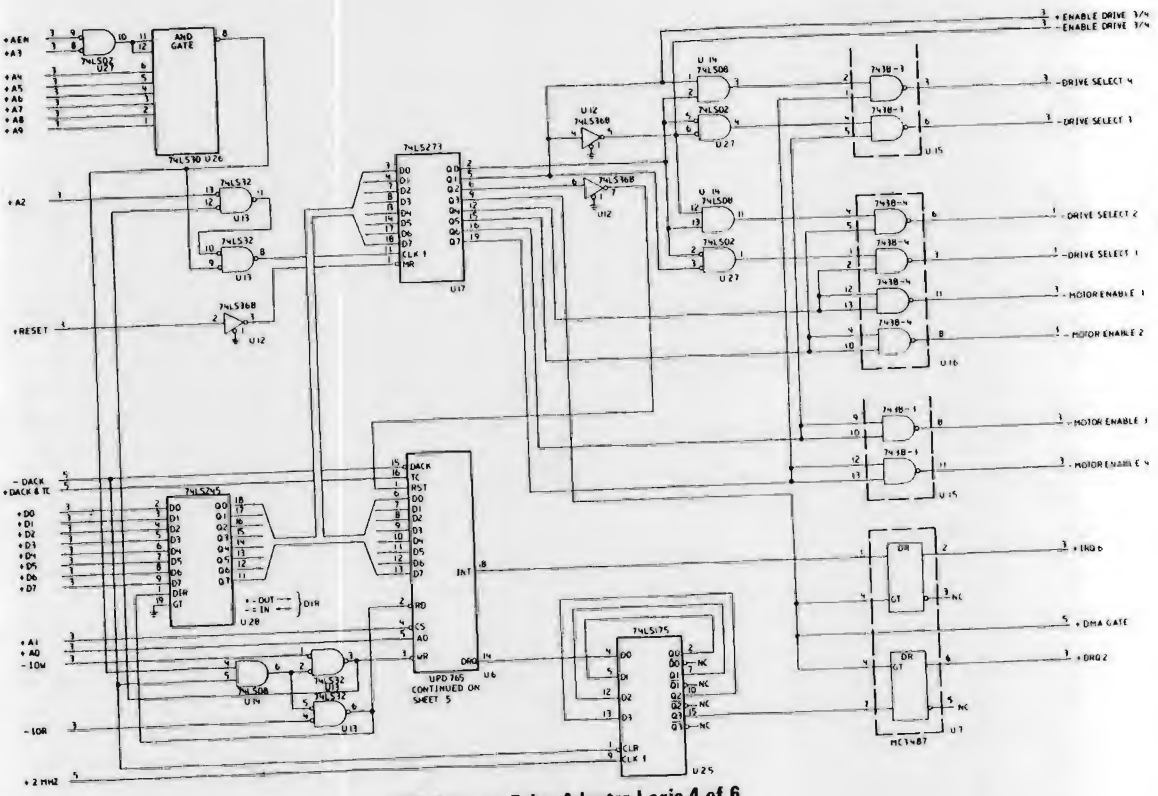


Parallel Printer Adapter

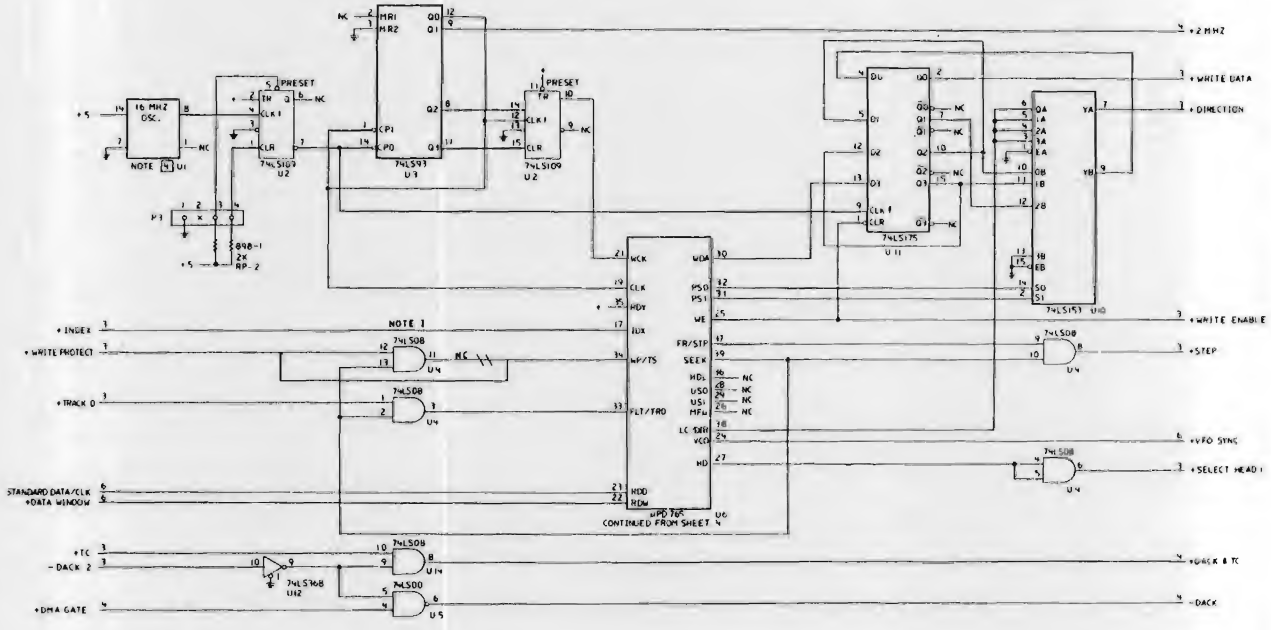


Note: Logics one and two of six are not applicable.

5¼" Diskette Drive Adapter Logic 3 of 6

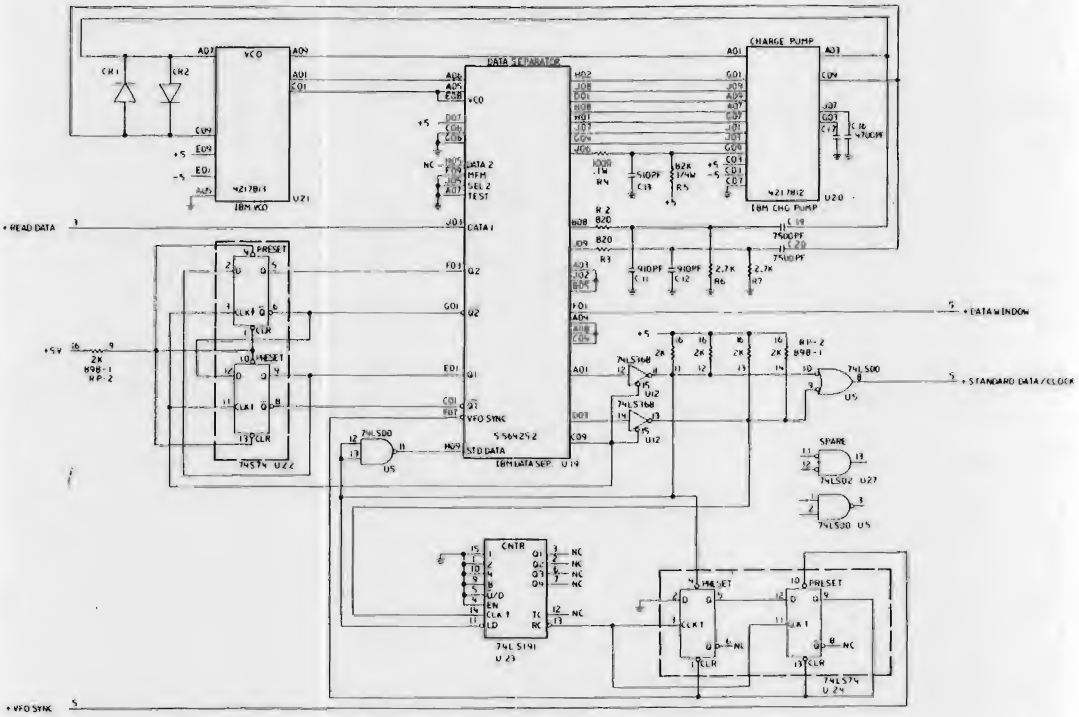


5¼" Diskette Drive Adapter Logic 4 of 6

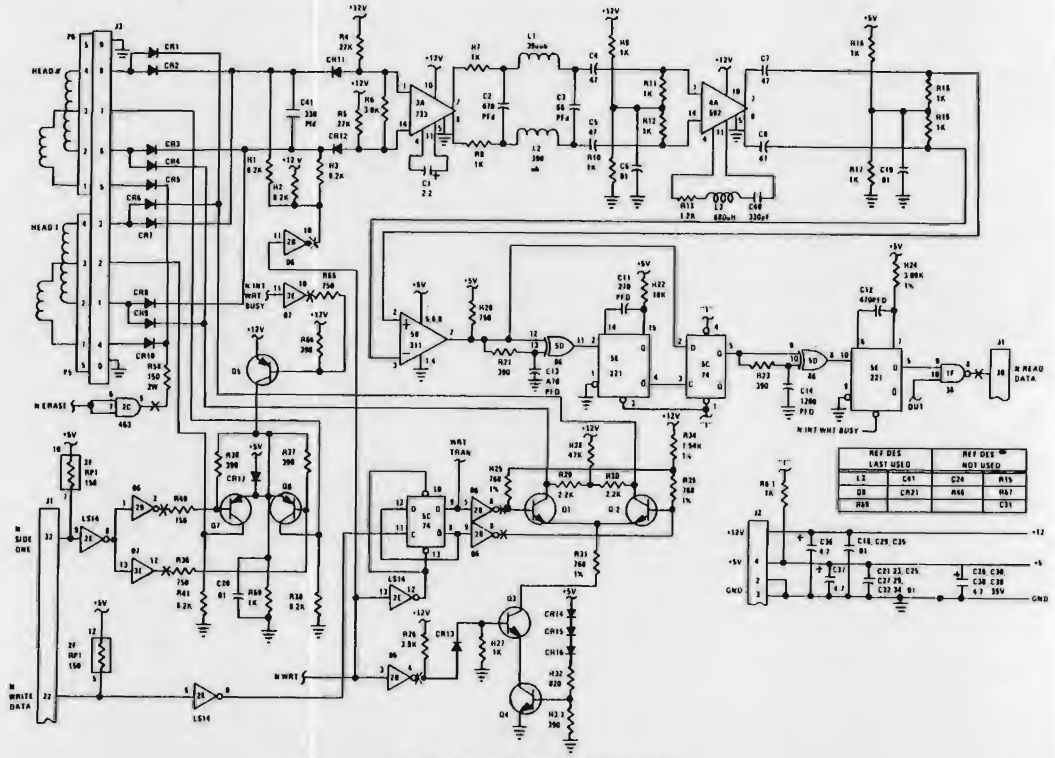


NOTE: U4 (74LS08) PINS 12 AND 13 ARE CONNECTED ONLY ON CARDS BUILT USING RAW CARD P/N 5001293

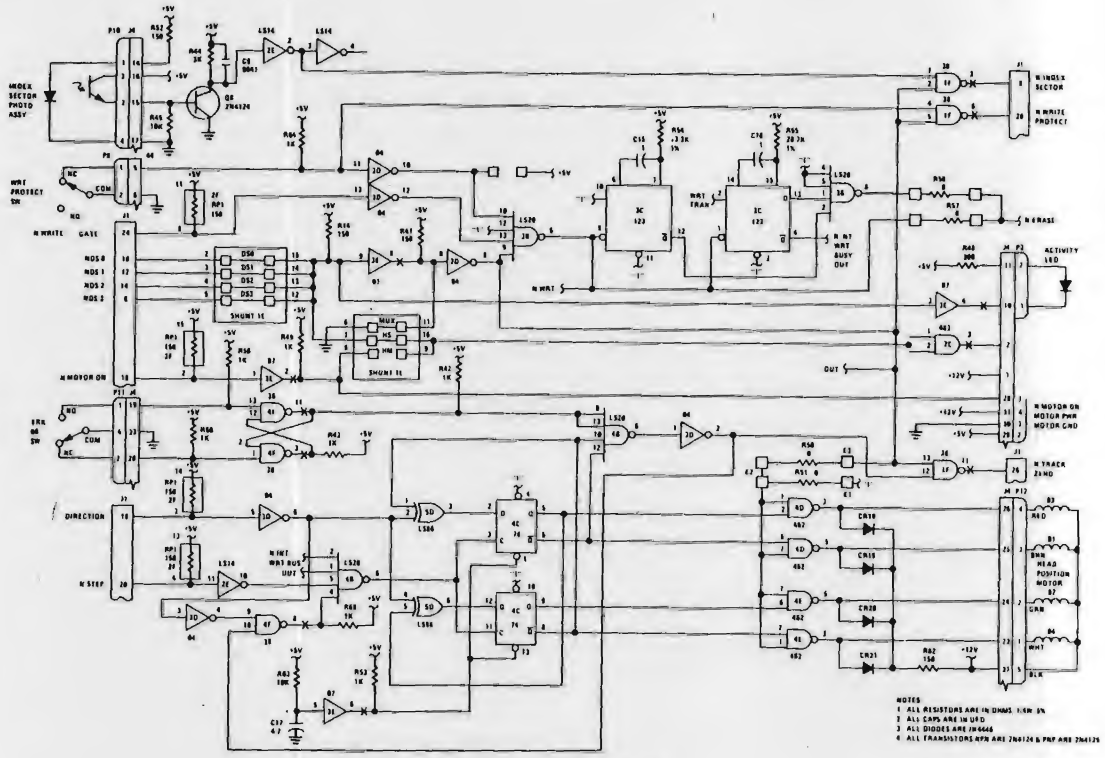
5¼" Diskette Drive Adapter Logic 5 of 6



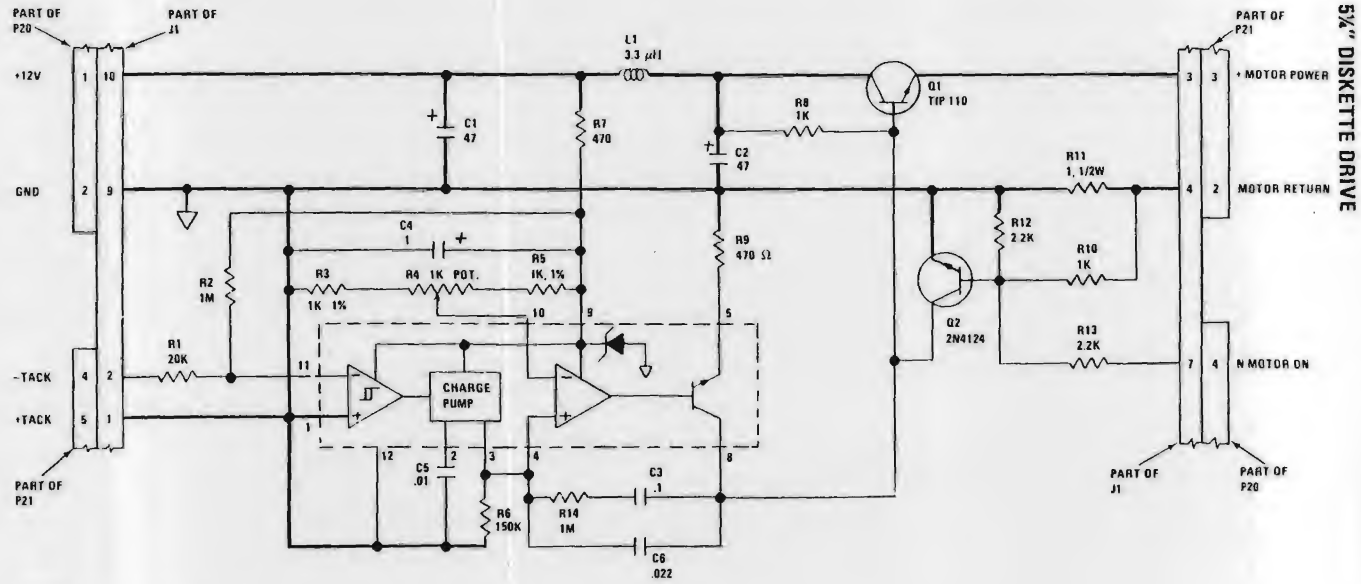
5¼" Diskette Drive Adapter Logic 6 of 6



5¼" Diskette Drive Logic 1 of 3

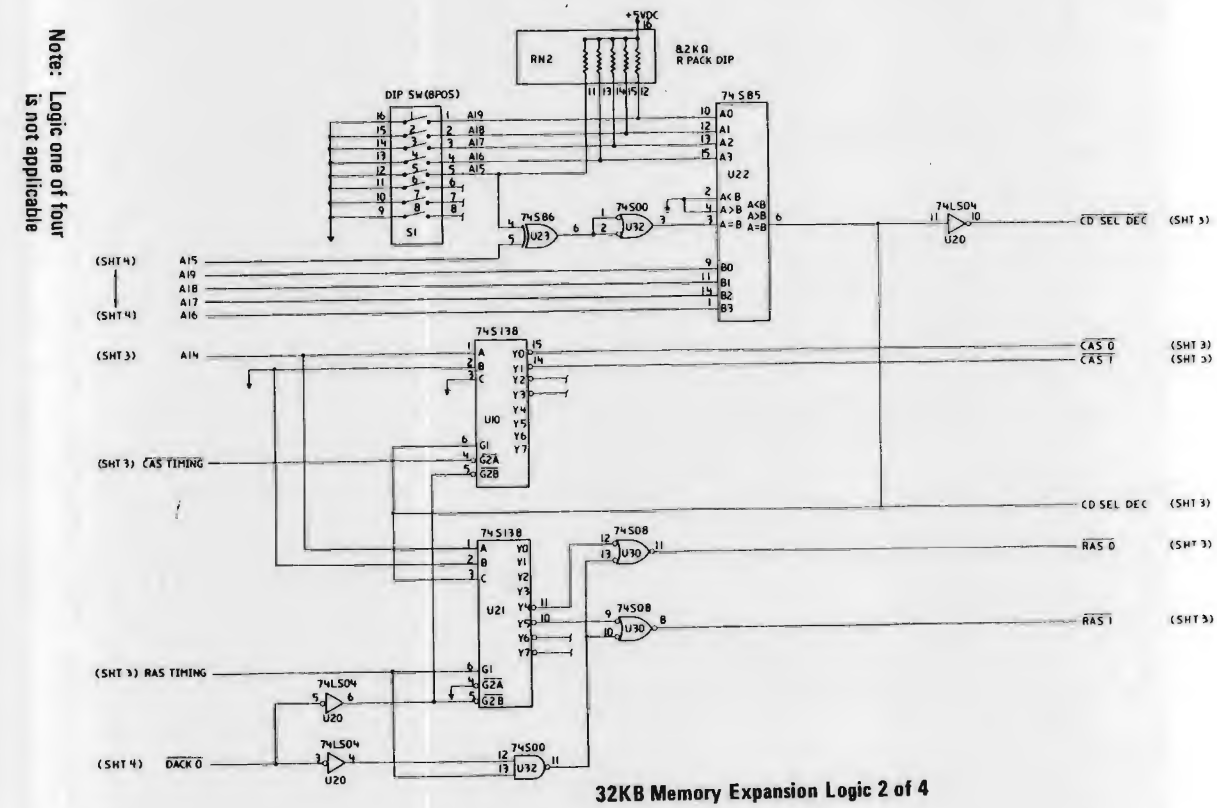


5¼" Diskette Drive Logic 2 of 3

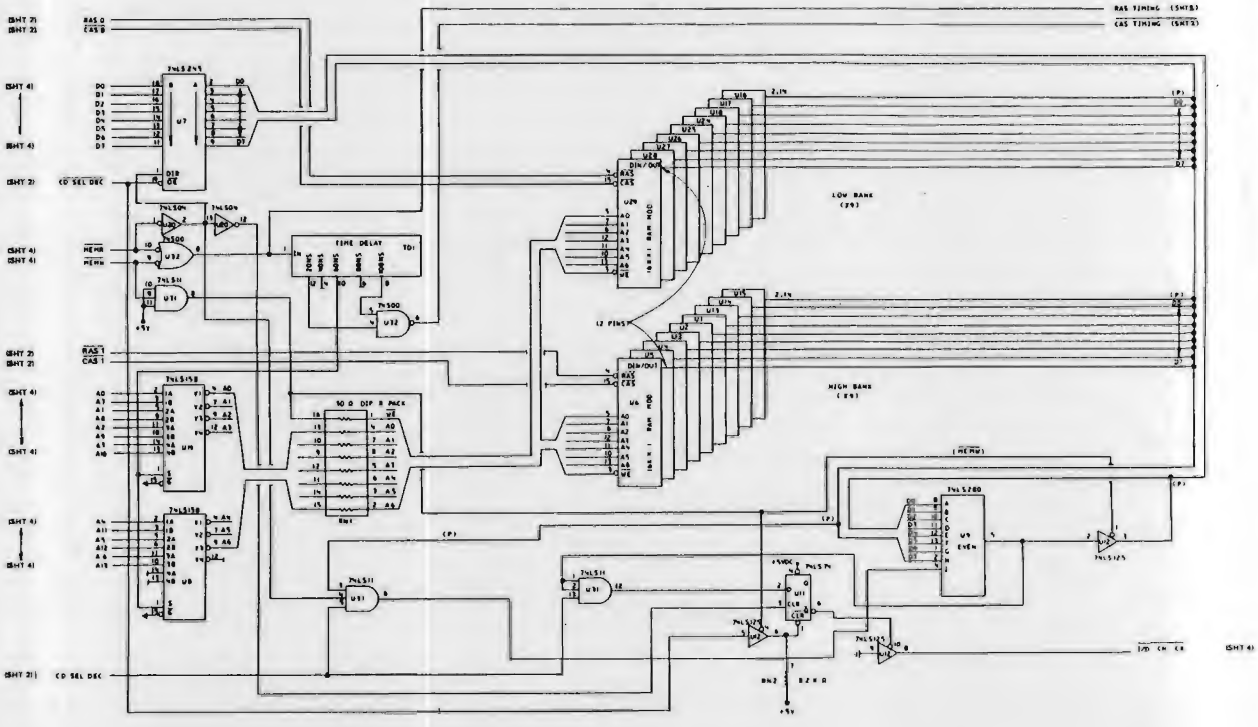


NOTES:
1. RESISTORS ARE IN OHMS, ± 5%, 1/4 W.
2. 1% RESISTORS ARE 1/8 W.
3. CAPACITORS ARE IN μ F, ± 20%, 35 V.

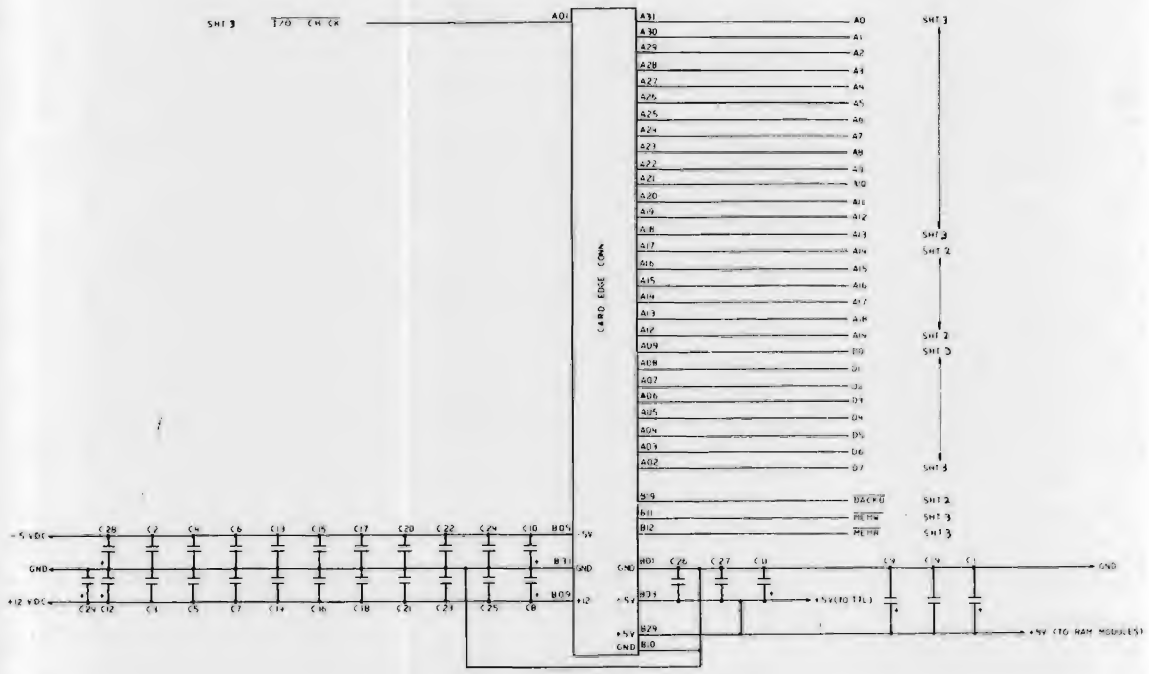
5¼" Diskette Drive Logic 3 of 3



32KB Memory Expansion Logic 2 of 4

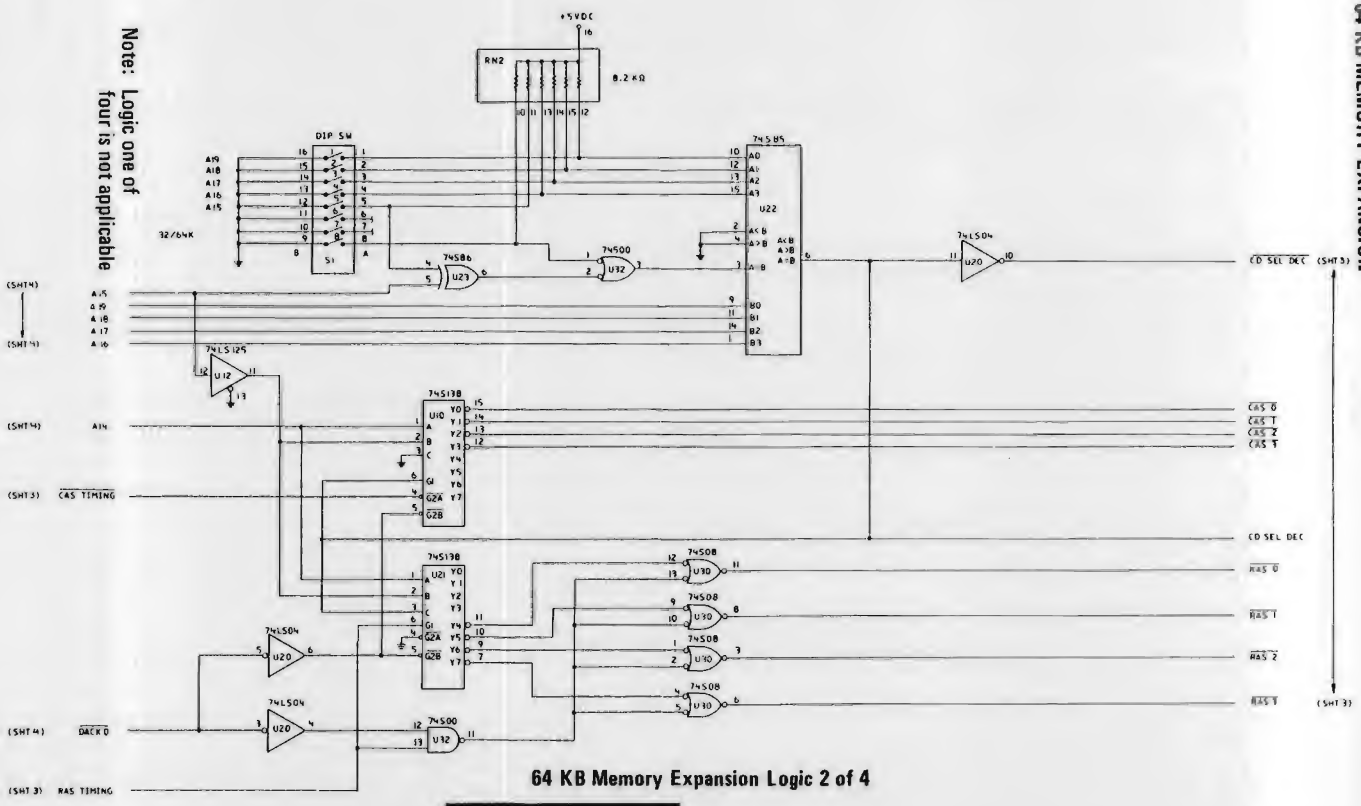


32KB Memory Expansion Logic 3 of 4

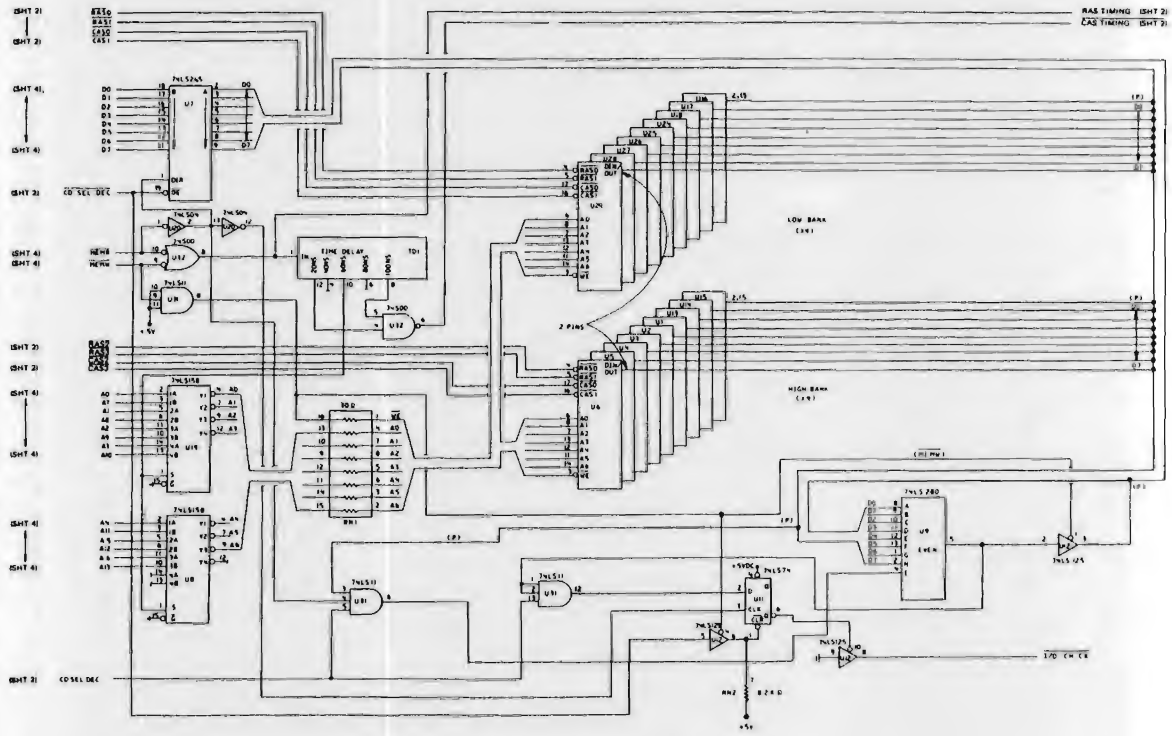


32KB Memory Expansion Logic 4 of 4

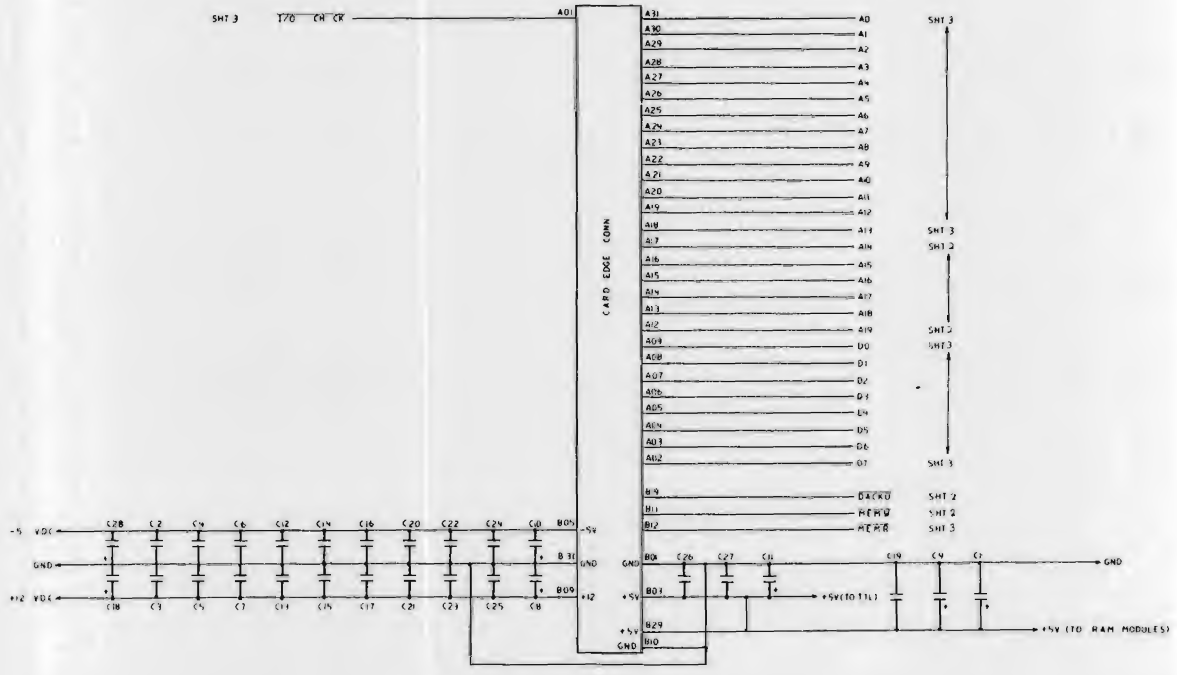
Note: Logic one of four is not applicable



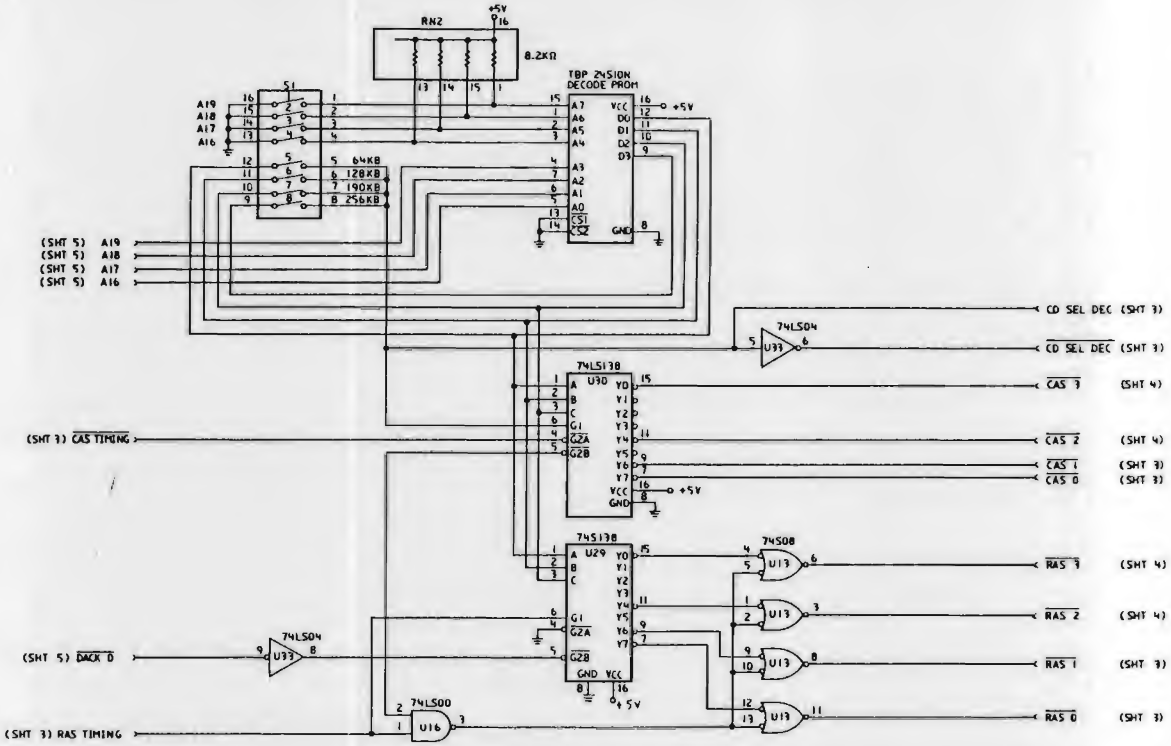
64 KB Memory Expansion Logic 2 of 4



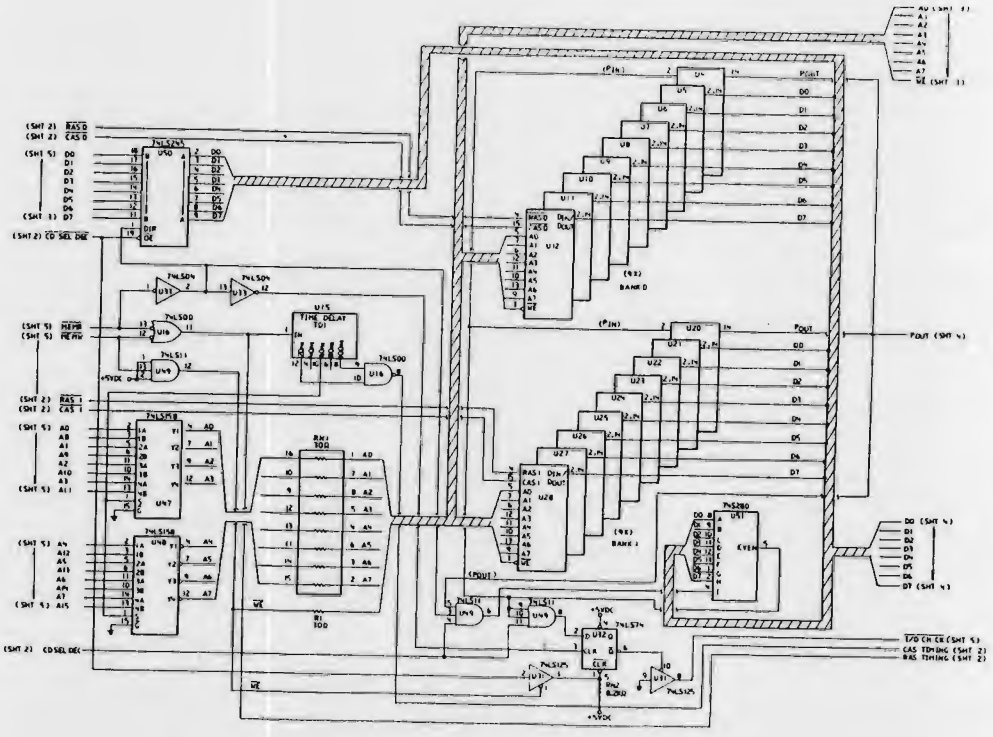
64KB Memory Expansion Logic 3 of 4



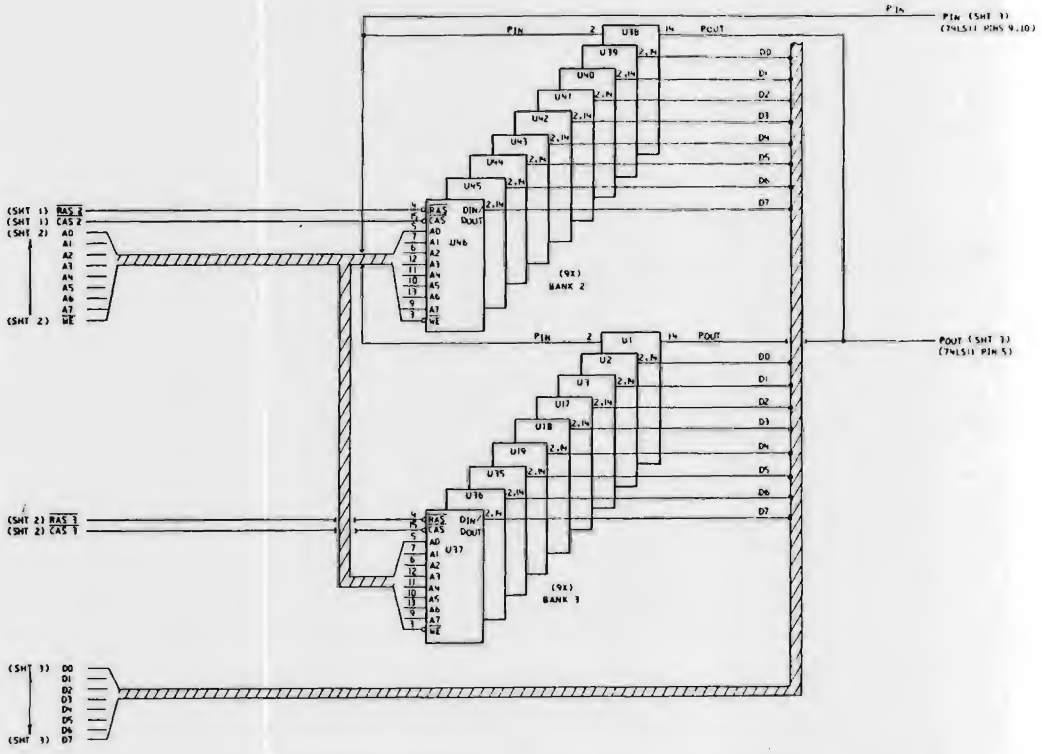
64 KB Memory Expansion Logic 4 of 4



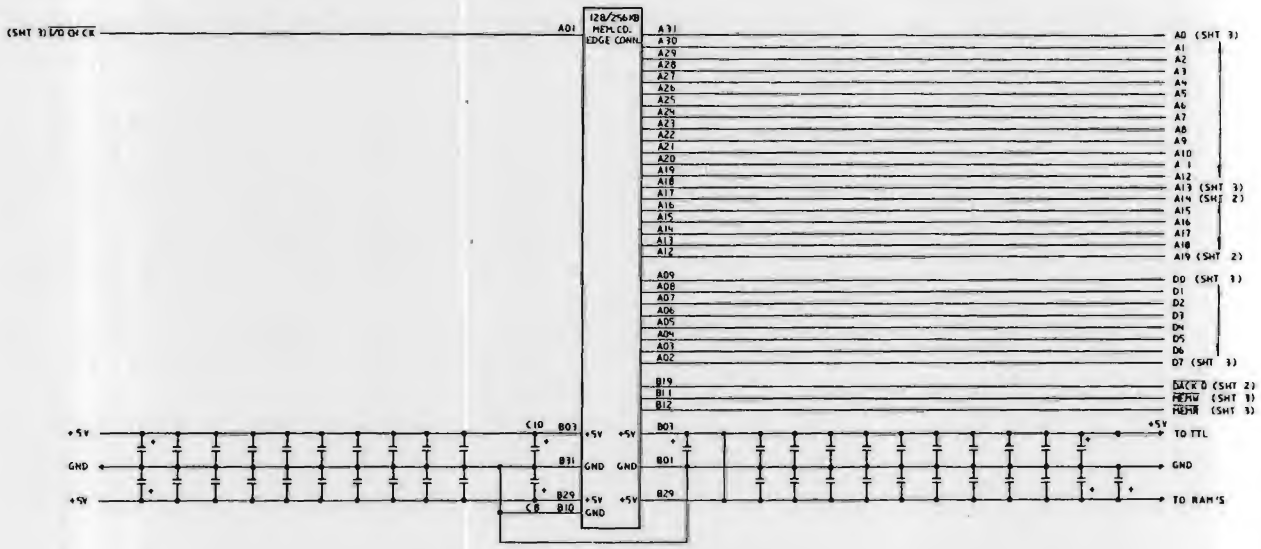
64/256KB Memory Expansion Logic 1 of 4



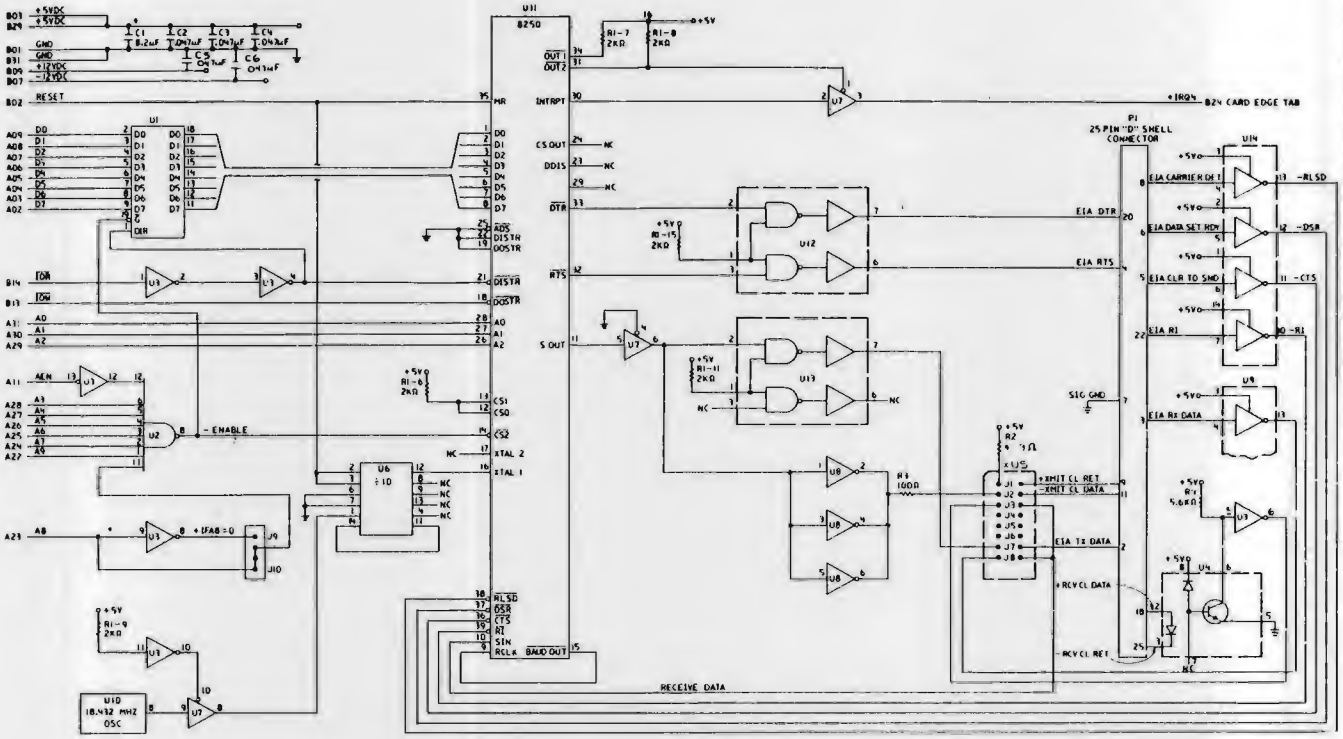
64/256KB Memory Expansion Logic 2 of 4



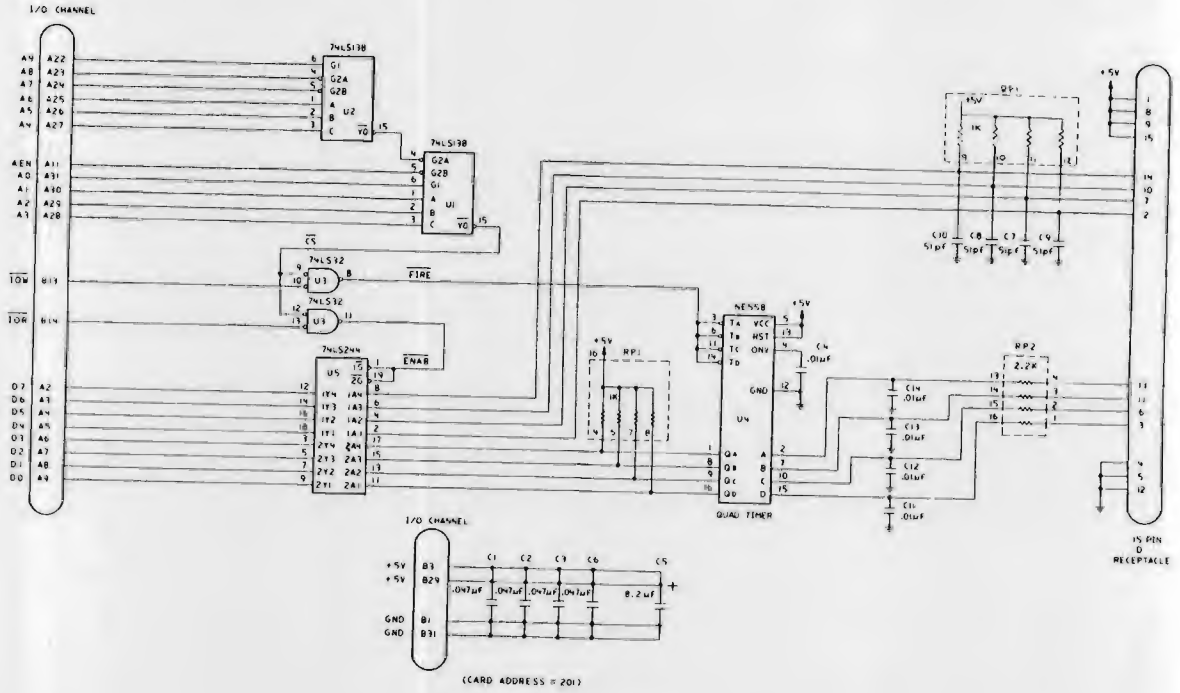
64/256KB Memory Expansion Logic 3 of 4



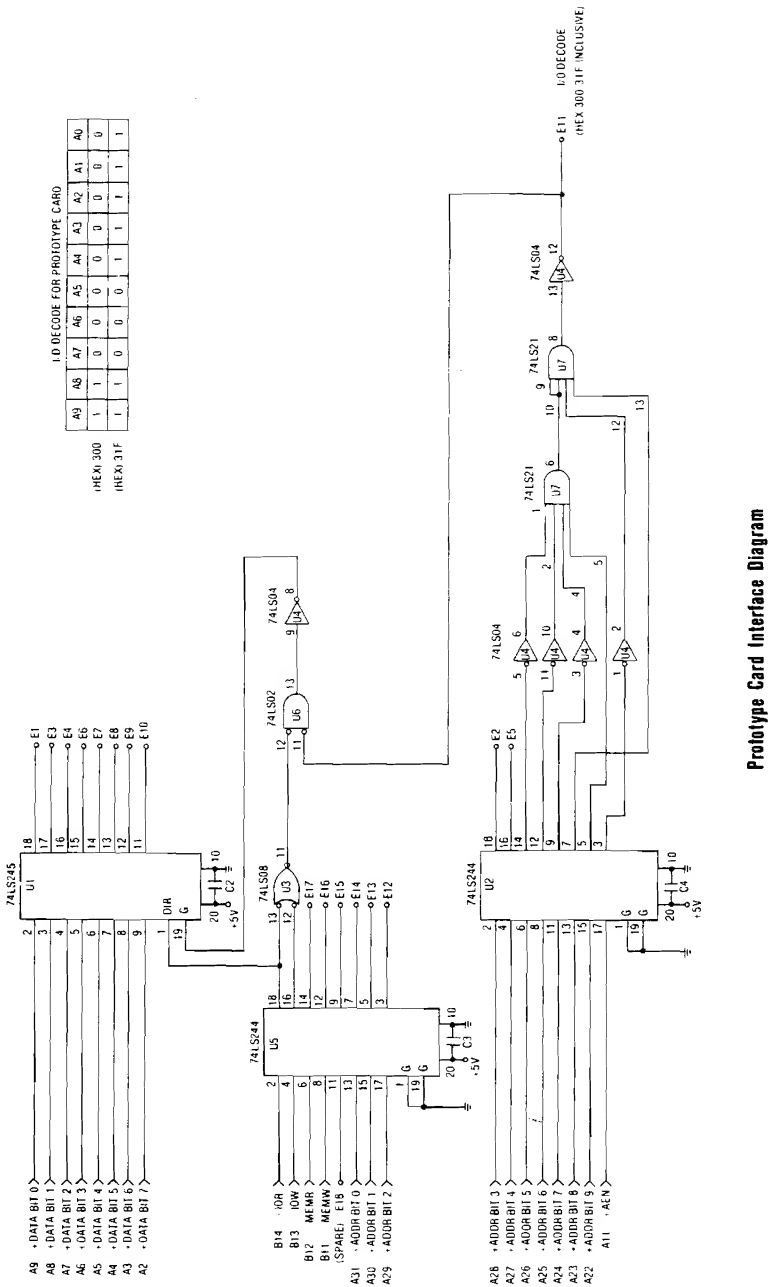
64/256KB Memory Expansion Logic 4 of 4



Asynchronous Communications Adapter



Game Control Adapter



Prototype Card Interface Diagram

APPENDIX E. UNIT SPECIFICATIONS

System Unit

Size:
- Length–19.6" (500 mm)
- Depth–16.1" (410 mm)
- Height–5.5" (142 mm)

Weight:
- Without Diskette Drive Unit-20.9 lbs (9.5 kg)
- With Diskette Drive Unit-25 lbs (11.4 kg)

Power Cable:
- Length–6'0" (1.83 m)
- Size–18 AWG

Environment:
- Air Temperature
  - System ON, 60° to 90° F (15.6° to 32.2° C)
  - System OFF, 50° to 110° F (10° to 43° C)
- Humidity
  - System ON, 8% to 80%
  - System OFF, 20% to 80%
- Heat Output, 1083 BTU/HR (Maximum)

Noise Levels:
- Without Printer, 59 DBS
- With Printer, 66 DBS

Electrical:

Nominal-120 VAC		220/240VAC
Minimum-104 VAC	OR	180 VAC
Maximum-127 VAC		259 VAC
KVA-.3175 maximum		KVA-.3175 maximum

Keyboard

Size:
- Length–19.6" (500 mm)
- Depth–7.87" (200 mm)
- Height–2.2" (57 mm)

Weight:
- 6.5 lbs (14.3 kg)

IBM Monochrome Display

Size:
- Length–14.9" (380 mm)
- Depth–13.7" (350 mm)
- Height–11" (280 mm)

Weight:
- 17.3 lbs (7.9 kg)

Heat Output:
- 325 BTU/HR

Power Cable:
- Length–3'0" (914 mm)
- Size–18 AWG

Signal Cable:
- Length–4'0" (1.22 m)
- Size–22 AWG

IBM 80 CPS Graphics Printer

Size:
- Width–14.7" (374 mm)
- Depth–12.0" (305 mm)
- Height–4.2" (107 mm)

Weight:
- 12.0 lbs (5.5 kg)

Power Cable:
- Length–6'0" (1.83 m)
- Size–18 AWG

Signal Cable:
- Length–6'0" (1.83 m)
- Size–22 AWG

Heat Output:
- 341 BTU/HR (Max.)

Electrical (three models available)

Nominal-	120 VAC	220 VAC	240 VAC
Minimum-	104 VAC	198 VAC	216 VAC
Maximum-	127 VAC	242 VAC	264 VAC

GLOSSARY

1. Address Buss: A set of wires or signals carrying the binary-coded address from the Intel-8088 microprocessor throughout the rest of the IBM Personal Computer System Unit.
2. AEN: Address Enable. (Refer to System Board I/O Channel Descriptions).
3. ALE: Address Latch Enable. (Refer to System Board I/O Channel Descriptions).
4. Analog: (1) Pertaining to representation by means of continuously variable physical quantities. (2) Contrast with digital.
5. A/N: Alphanumeric: Pertaining to a character set that contains letters, digits, and usually other characters, such as punctuation marks. Synonymous with alphameric.
6. A0-A19: Address bits 0-19. (Refer to System Board I/O Channel Descriptions).
7. APA: All points addressable graphics.
8. ASCII: American Standard Code of Information Interchange. The standard code, using a coded character set consisting of 7-bit coded characters (8 bits including parity check), used for information interchange among data processing systems, data communication systems and associated equipment. The ASCII set consists of control characters and graphic characters.
9. Assembler: A computer program used to assemble. Synonymous with assembly program.
10. BASIC: (Beginner's all-purpose symbolic instruction code). A programming language with a small repetoire of commands and a simple syntax, primarily designed for numerical application.
11. BAUD: (1) A unit of signaling speed equal to the number of discrete conditions or signal events per second in Morse code, one bit per second in a train of binary signals, and one 3-bit value per second in a train of signals each of which can assume one of eight different states. (2) In asynchronous transmission, the unit of modulation rate corresponding to one unit of interval per second, i.e. if the duration of the unit interval is 20 milliseconds, the modulation rate is 50 baud.

12. Binary: (1) Pertaining to a selection, choice, or condition that has two possible values or states. (2) Pertaining to a fixed radix numeration system having a radix of two.
13. BIOS: Basic Input/Output System.
14. Bootstrap: A technique or device designed to bring itself into a desired state by means of its own action, e.g. a machine routine whose first few instructions are sufficient to bring the rest of itself into the computer from an input device.
15. Buffer: An area of storage that is temporarily reserved for use in performing an input/output operation, into which data is read or from which data is written. Synonymous with I/O area. A portion of storage for temporarily holding input or output data.
16. Bus: One or more conductors used for transmitting signals or power.
17. Byte: (1) A binary character operated upon as a unit and usually shorter than a computer word. (2) The representation of a character.
18. CLK: Clock. (Refer to System Board I/O Channel Descriptions).
19. Code: (1) A set of unambiguous rules specifying the manner in which data may be represented in a discrete form. Synonymous with coding scheme. (2) A set of items such as abbreviations representing the members of another set. (3) Loosely, one or more computer programs, or part of a computer program. (4) To represent data or a computer program in a symbolic form that can be accepted by a data processor.
20. Computer: A data processor that can perform substantial computation, including numerous arithmetic operations, or logic operations, without intervention by a human operator during the run.
21. CPS: Characters per second.
22. CRC: The cyclic redundancy check character.
23. CRT: (1) A Cathode ray tube display. (2) A display device, such as the IBM Monochrome Display, that uses a cathode ray tube.
24. CTS: Conversational Terminal System. (2) Clear to Send. Associated with modem control.
25. DACK0-DACK3: DMA Acknowledge 0 to 3. (Refer to System Board I/O Channel Description.)

26. Data: (1) A representation of facts, concepts or instructions in a formalized manner suitable for communication, interpretation, or processing by humans or automatic means. (2) Any representations such as characters or analog quantities to which meaning is, or might be assigned.
27. Din Connectors: One of the connectors specified by the Din standardization committee.
28. DIP: "Dual In-Line Package." A widely used container for an integrated circuit. DIP's are pins usually in two parallel rows. These pins are spaced on 1/10" inters and come in different configurations ranging from a 14-pin assembly to a 40-pin configuration.
29. Display: A visual presentation of data.
30. DMA: Direct Memory Access.
31. DO-D7: Data Bits 0 to 7. (Refer to System Board I/O Channel Descriptions.)
32. DRQ1-DRQ3: DMA Request 1 to 3. (Refer to System Board I/O Channel Descriptions.)
33. DSR: Data Set Ready, associated with modem control.
34. DTR: Distribution Tape Reel.
35. Edge Connector: An opening which joins with the end of a circuit board. The purpose of this interface is to send electrical signals back and forth.
36. EIA/CCITT Drives: Electronic Industries Association/ Consultative Committee on International Telegraphy and Telephony Drives.
37. EPROM or 'PROM': Term for "Programmable Read-Only Memory." An EPROM or 'PROM' is actually Read-Only Memory (ROM) but the contents may be changed by electrical means. EPROM or 'PROM' information is not destroyed when the power is cut off.
38. Firmware: Memory chips with the software programs already built in.
39. Graphics: Symbols Produced by a process such as handwriting, drawing or printing. Synonymous with graphic symbol.
40. Hexadecimal: Pertaining to a selection, choice, or condition that has sixteen possible values or states. These values or states usually contain 10 digits and six letters A through F. Hexadecimal digits are equivalent to a power of 16.

41. Hertz (Hz): A unit of frequency equal to one cycle per second.
42. High order position: The leftmost position in a string of characters.
43. Input/Output (I/O): Pertaining to a device or to a channel that may be involved in an input process, and, at a different time, in an output process. (2) Pertaining to a device whose parts can be performing an input process and an output process at the same time.
44. Integrated Circuit: A combination of interconnected circuit elements inseparably associated on or within a continuous substrate.
45. Interpreter: A computer program used to interpret. Synonymous with interpretive program.
46. Interrupt: (1) A suspension of a process, such as the execution of a computer program, in such a way that the process can be resumed. (2) To stop a process in such a way that it can be resumed. (3) In data transmission, to take an action at a receiving station that causes the transmitting station to terminate a transmission.
47. I/O Channel: Input/Output Channel. In a data processing system, a functional unit, controlled by the processing unit, that handles the transfer of data between main storage and peripheral equipment.
48. I/O CH CK: I/O Channel Check. (Refer to System Board I/O Channel Descriptions.)
49. I/O CH RDY: I/O Channel Ready. (Refer to System Board I/O Channel Descriptions.)
50. IMR: Interruption Mask Register.
51. IOR: I/O Read Command. (Refer to System Board I/O Channel Descriptions.)
52. IOW: I/O Write Command. (Refer to System Board I/O Channel Descriptions.)
53. IRQ2-IRQ7: Interrupt Request 2 to 7. (Refer to System Board I/O Channel Descriptions.)
54. K: An abbreviation for the prefix kilo, i.e. 1000 in decimal notation. To the tenth power, 1024 in decimal notation.
55. KB: Kilobyte.
56. Khz: Kilohertz. A unit of frequency equal to 1,000 hertz.

57. Low order position: The rightmost position in a string of characters.
58. Machine Language: (1) A language that is used directly by a machine. (2) Another term for computer instruction code.
59. Memory Address: A two-byte value selecting one specific memory location on a memory map.
60. Memory Location: The most specific part of a memory map that the computer can refer to.
61. Memory Map: The list of memory locations addressed directly by the microprocessor.
62. MEMR: Memory Read Command. (Refer to System Board I/O Channel Descriptions.)
63. MEMW: Memory Write Command. (Refer to System Board I/O Channel Descriptions.)
64. MFM Coded: Modified Frequency Modulation. It is double density encoding of information on a diskette.
65. Mhz: Megahertz. A unit of frequency equal to one million Hertz.
66. Microprocessor: A processing unit, or part of a processing unit, that consists of microcode. In the IBM Personal Computer, the microprocessor is the Intel-8088.
67. Mnemonic: Symbol or symbols used instead of terminology more difficult to remember. Usually a mnemonic has two or three letters.
68. Mode: (1) A method of operation, for example, the binary mode, the interpretive mode, the alphanumeric mode. (2) The most frequent value in the statistical sense.
69. Monitor: (1) A device that observes and verifies the operation of a data processing system and indicates any specific departure from the norm. (2) A television type display such as the IBM Monochrome Display. (3) Software or hardware that observes, supervises, controls, or verifies the operations of a system.
70. Multiplexer: A device capable of interleaving the events of two or more activities or capable of distributing the events of an interleaved sequence to their respective activities.

71. OR: A logic operator having the property that if P is a statement, Q is a statement, R is a statement..., then the OR of P,Q,R, is true if at least one statement is true, false if all statements are false. P OR Q is often represented by P+Q, PVQ. The term is synonymous with boolean add; logic add.
72. "ORed": Past tense of OR.
73. OSC: Oscillator. (Refer to System Board I/O Channel Descriptions.)
74. Output: Pertaining to a device, process, or channel involved in an output process, or to the data or states involved in an output process.
75. Personal Computer: A small home or business computer complete with a System Unit, keyboard, and available with a variety of options such as monochrome display and a dot matrix printer.
76. Pinout: A diagram of functioning pins on a pinboard.
77. Printed Circuit Board: A piece of material, usually fiberglass, which contains a layer of conductive material, usually metal. The metallic layer is then etched and electronic equipment is then attached to the fiberglass. The electronic equipment then has the capacity to transmit electronic signals through the board by way of the etched metal tracks.
78. Program: (1) A series of actions designed to achieve a certain result. (2) To design, write and test computer programs.
79. Read/Write Memory: Random access storage.
80. Reset Drv: Reset Driver. (Refer to System Board I/O Channel Descriptions.)
81. RF Modulator: The device used to convert the composite video signal to the antenna level input of a home TV.
82. ROM: Read-only Memory.
83. ROM BIOS: Read-only Memory/Basic Input Output System.
84. RS 232 Port: Asynchronous Type Communications.
85. RTS: Ready to Send. Associated with modem control.

86. Scan Line: The use of a cathode beam to test the cathode ray tube of a display used with a personal computer.
87. Schematic: The description, usually in diagram form, of the logical structure and physical structure of an entire data base according to a conceptual model.
88. Software: (1) Computer programs, procedures, rules, and possibly associated documentation concerned with the operation of a data processing system. (2) Contrast with hardware.
89. Strobe: (1) An instrument used to determine the exact speed of circular or cyclic movement. (2) A flashing signal displaying an exact event.
90. Text: In ASCII and data communication, a sequence of characters treated as an entity if preceded and terminated by one STX and one ETX transmission control respectively.
91. TX Data: Transmit Data. External connections of the RS 232 Asynchronous Communications Adapter interface.
92. Video: Computer data shown or displayed on a cathode ray tube monitor or display.

NOTES

BIBLIOGRAPHY

IBM Publications

1. IBM Personal Computer Guide to Operations
   General information on using the IBM Personal Computer.
2. IBM Personal Computer Hardware and Service
   Information on hardware and steps necessary when servicing the IBM Personal Computer.
3. IBM Personal Computer BASIC
   Information for programmers who are using BASIC.
4. IBM Personal Computer Disk Operating System (DOS)
   Information for programmers who are using DOS.
5. IBM Personal Computer MACRO Assembler
   Information for experienced assembly language programmers using the Macro Assembler.
6. IBM Personal Computer Pascal Compiler
   Information for programmers who are familiar with the Pascal language.

Other Related Publications

7. NATIONAL SEMICONDUCTOR
   INS 8250 Asynchronous Communications Element
   This book documents physical and operating characteristics of the INS 8250.
8. INTEL
   The 8086 Family Users Manual
   This manual introduces the 8086 family of microcomputing components and serves as a reference in system design and implementation.
9. INTEL
   8086/8087/8088 Macro
   ASSEMBLY Language
   Reference Manual for 8088/8085 Based Development System
   The manual describes the 8086/8087/8088 Macro Assembly Language, and is intended for use by persons who are familiar with assembly language.
10. MOTOROLA
    The complete Microcomputer Data Library
    This book can provide additional information on the Motorola 6845 CRT Controller used in the IBM Monochrome Display and Parallel Printer Adapter, and the Color/Graphics Monitor Adapter.

INDEX

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W

Numerics